# Relational Databases and Anomalies

.NET

*A Relational Database is made up of a collection of **tables** that each store a specific set of structured data.*
*A table contains a collection of rows (**tuples**) and columns (**attributes**). Each column in the table stores a certain type of data.*

HTTPS://DOCS.MICROSOFT.COM/EN-US/SQL/RELATIONAL-DATABASES/DATABASES/DATABASES?VIEW=SQL-SERVER-VER15

# Databases – Instances/Users

https://docs.microsoft.com/en-us/sql/relational-databases/databases/databases?view=sql-server-ver15

There are one or more *schemas* within a database. Within each *schema* there are database objects such as *tables* and *views*.

This is a *table*. →→→→→



Foreign Key

A row(Tuple) makes up an entity

Primary Key

Attributes / Candidate Keys

| | Name | Position | Office | Age | Start date | Salary |
|---|---|---|---|---|---|---|
| | Tiger Nixon | System Architect | Edinburgh | 61 | 2011/04/25 | $320,800 |
| | Garrett Winters | Accountant | Tokyo | 63 | 2011/07/25 | $170,750 |
| | Ashton Cox | Junior Technical Author | San Francisco | 66 | 2009/01/12 | $86,000 |
| | Cedric Kelly | Senior Javascript Developer | Edinburgh | 22 | 2012/03/29 | $433,060 |
| | Airi Satou | Accountant | Tokyo | 33 | 2008/11/28 | $162,700 |
| | Name | Position | Office | Age | Start date | Salary |

Show 5 entries    Search:

Showing 1 to 5 of 57 entries    Previous  1  2  3  4  5  ...  12  Next

# DBMS (Database Management System)

- <u>Data</u> is a collection of facts and figures that can be processed to produce information.

- A <u>Database</u> is a collection of related data.

- A *DBMS* stores data so that it is easier to retrieve and insert.

If we have data about grades of all students, we can extrapolate average grades and highest grades.

Primary Key ➡

| Last Name | Birthday | Grade 1 | Grade 2 | Grade 3 | Final Grade |
|-----------|----------|---------|---------|---------|-------------|
| Smith | June 7, 1987 | 100 | 98 | 89 | 95.67 |
| Jones | October 5, 1986 | 75 | 89 | 84 | 82.67 |
| Garcia | December 15, 1986 | 99 | 97 | 100 | 98.67 |
| Kim | February 28, 1987 | 50 | 68 | 42 | 53.33 |
| Washington | May 4, 1987 | 85 | 87 | 79 | 83.67 |
| Hernandez | October 8, 1986 | 74 | 72 | 81 | 75.67 |
| Gates | March 21, 1987 | 32 | 54 | 67 | 51.00 |
| Papert | April 26, 1987 | 84 | 92 | 81 | 85.67 |
| Kennedy-Onassis | January 18, 1987 | 92 | 90 | 88 | 90.00 |
| Smith | February 8, 1987 | 72 | 65 | 99 | 78.67 |

# DBMS - Benefits

Relation-based tables − A Relational DBMS allows *entities* and *relations* among them to form *tables*.

Separation of data and application − A database is separated from its data. A database is an active entity, whereas data on which the database works is said to be passive.

Less redundancy − DBMS follows the rules of *normalization*, which splits a relation when any of its attributes has redundancy.

*Consistency* − Consistency is the state where every relation in a database is *persisted*.

Query Language − DBMS (or RDBMS) is equipped with query language (usually *SQL*), which makes it more efficient to retrieve and manipulate data.

*ACID Properties* − A DBMS follows the concepts of *Atomicity*, *Consistency*, *Isolation*, and *Durability*.

*Isolation Levels* − A DBMS supports a multi-user environment and applies restrictions on the access and manipulation of data in parallel.
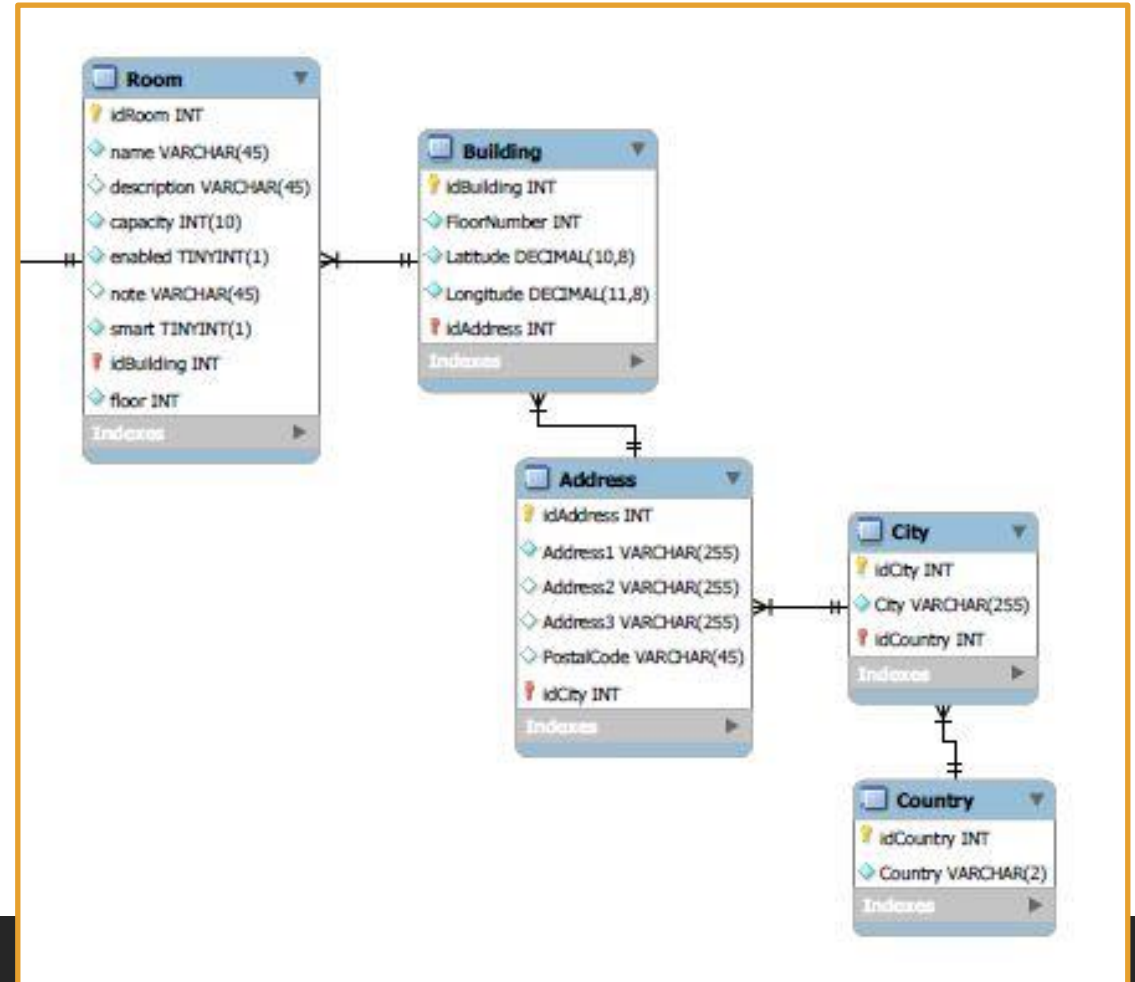
# Database – Schema Diagram

A database *schema* is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and the *relations* among them.

It displays all the *constraints* that are to be applied on the data.

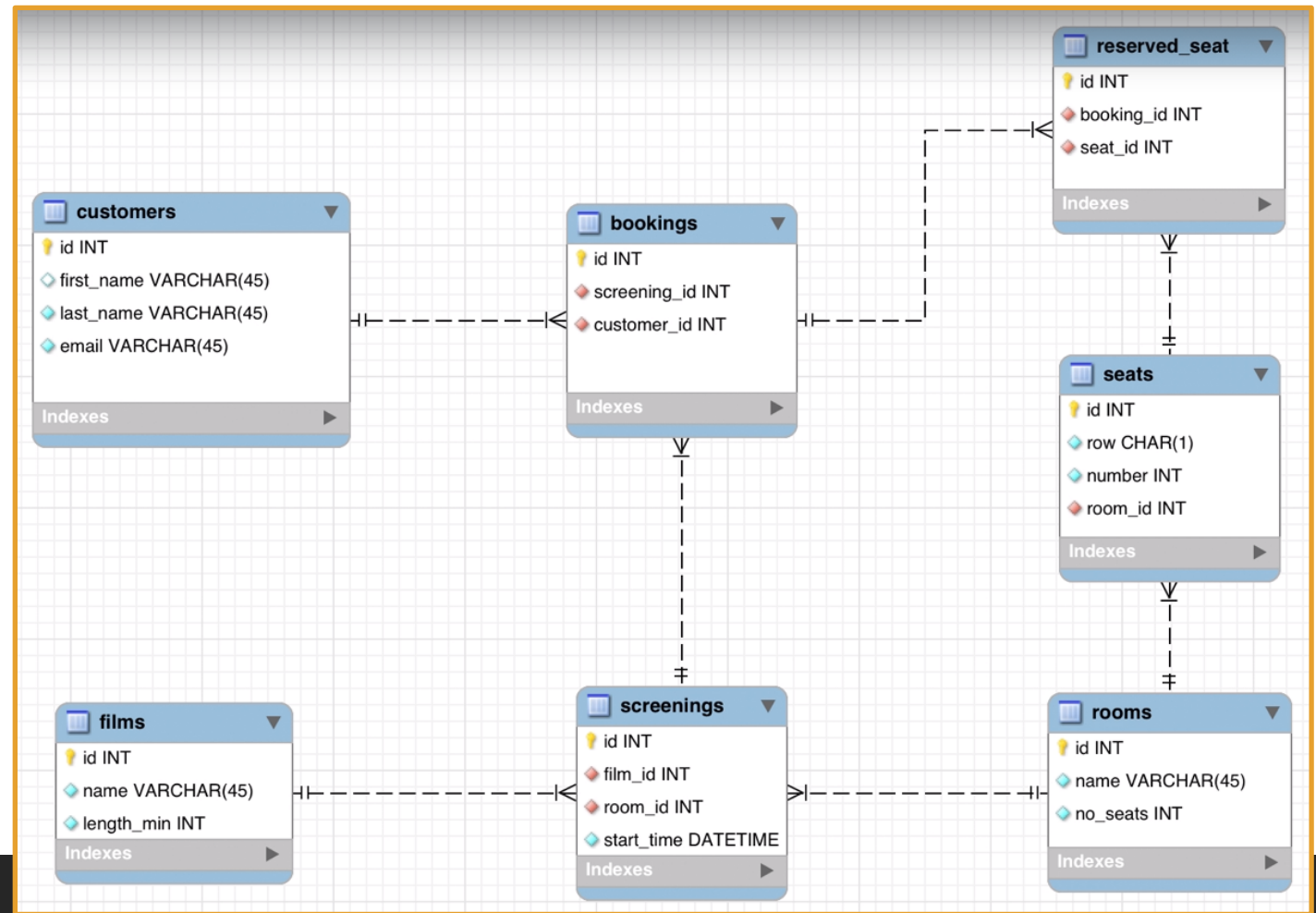A *schema diagram* contains a descriptive detail of the database.

# Database – Schema Diagram

https://www.tutorialspoint.com/dbms/dbms_data_schemas.htm

**Red Color means FK**

◆◆ Solid Shape == NOT NULL Constraint

◇◇ Empty Color Outline == NULL

🔑 Golden key == PK

🔑 Red Key == PK also FK

◆ Blue Diamond == NOT NULL Constraint

◆ Red Diamond == NOT NULL and is a FK.

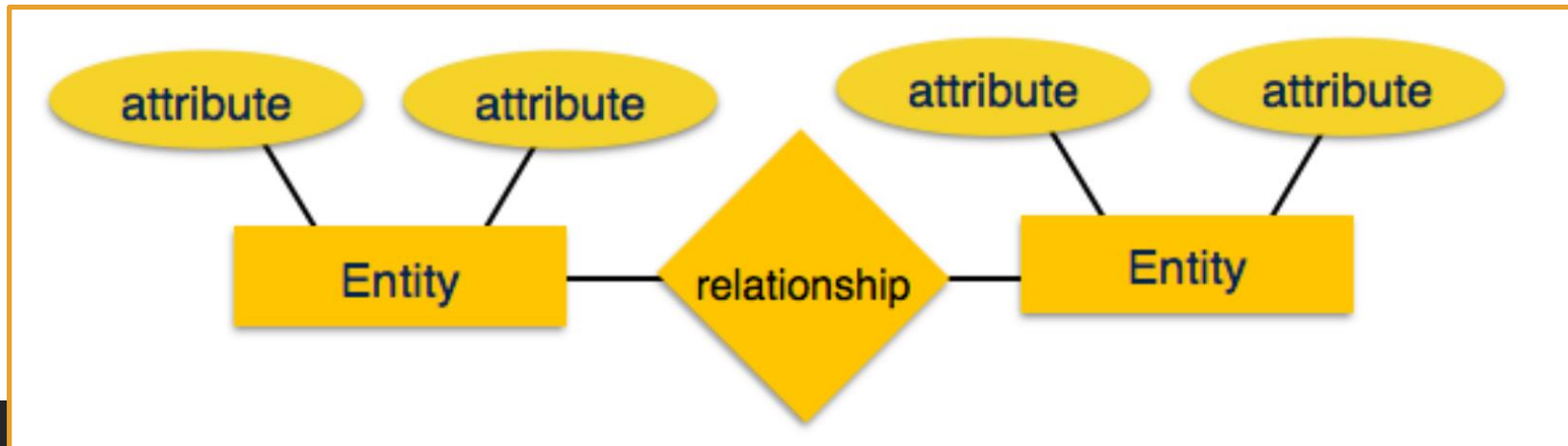# Database – Entity-Relationship Models

*Data models* define the structure of a database. *Data Models* are *entities* that abstract the data from a *DBMS* into a more human readable format. *Data models* define how data is connected to each other and how they are processed and stored inside the database.
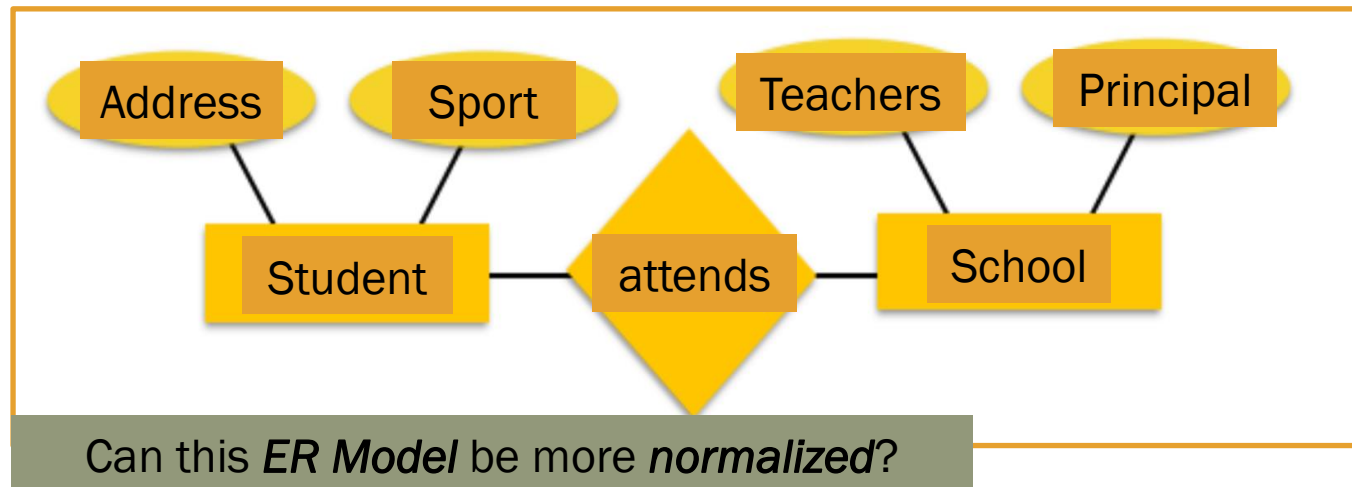
An *Entity-Relationship (ER) Model* is based on the notion of real-world entities and the relationships between them. An *ER Model* is used for the conceptual design of a database.

# Database – Entity-Relationship Models
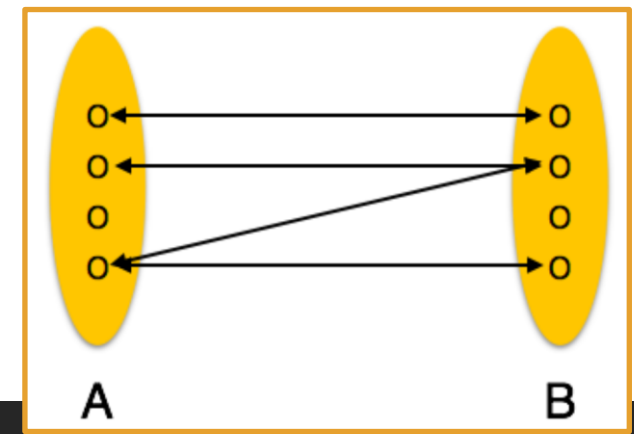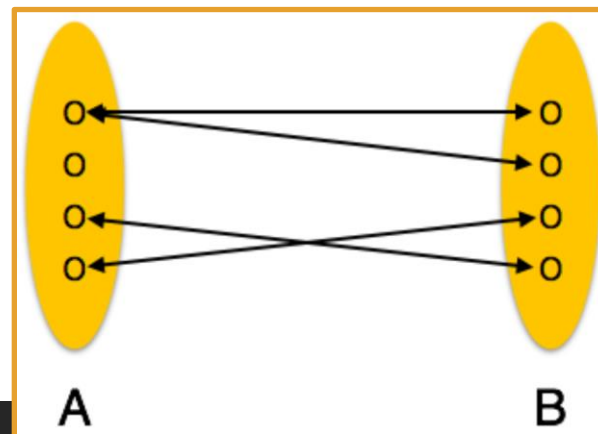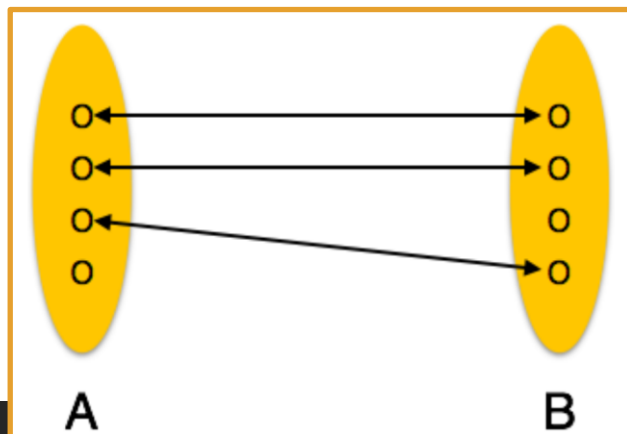
https://www.tutorialspoint.com/dbms/dbms_data_models.htm

- **Entity** – a real-world thing having properties called **attributes**.
- **Attribute** - the details about an *entity*.
- **Relationship** – The logical association among entities. Relationships are mapped with entities in various ways.
- "Mapping cardinalities" defines the number of associations between two entities.

- Mapping cardinalities
  - *one to one*
  - *one to many*
  - *many to many*



Can this *ER Model* be more *normalized*?

# Database – Relationships

| Cardinality | | |
|---|---|---|
| One-to-one | One-to-many | Many-to-many |
| One *entity* from *entity* set A can be associated with at most one *entity* of *entity* set B and vice versa. | One *entity* from *entity* set A can be associated with more than one *entities* of *entity* set B. However, an *entity* from *entity* set B can be associated with at most one other *entity* | One *entity* from A can be associated with more than one *entity* from B and vice versa. Use a Cross-Reference Table. |

# Relational Databases – Concepts

_Tables_ – "relations" are saved in table format. This format stores the relation among _entities_. A table has rows and columns, where rows represent _entities_ and columns represent _attributes_.

_Tuple_ – A single row of a _table_, which contains a single record for that _entity_ is called a _tuple_.

_Relation schema_ – A relation schema describes the relation name (table name), attributes, and their names.

_Primary Key_ – Each row has one (or more) attributes, chosen as _Primary keys_. These identify the row in the _table_ uniquely.

_Composite Key_ – Multiple _Candidate Keys_ that together form the _Primary Key_.

_Index_ – A unique number given to each _tuple_ in a _table_ to serve as the _Primary Key_.

# Databases –
# Primary and Candidate Keys

https://www.tutorialspoint.com/dbms/er_model_basic_concepts.htm
https://docs.microsoft.com/en-us/ef/core/modeling/keys?tabs=data-annotations

*Candidate Key* – An *attribute* of an entity. An *entity* set may have more than one *candidate key*.

*Primary Key* – A *Candidate Key* chosen to uniquely identify the *entity* set(*tuple*).

*Foreign Key* – The *Primary Key* of another table. Used to represent to other entity.



Foreign Key

A Tuple makes up an entity

Primary Key

Attributes / Candidate Keys

# Relational Databases

A *RDBMS* allows *entities* to form *tables* with *relations* between them. You could understand the architecture of a database just by looking at the *table* names.



**Students : Table**

| | | StudentID | LastName | FirstName | State | Zip |
|---|---|---|---|---|---|---|
| ▶ | + | 151 | Galla | Kristi | NJ | 19038 |
| | + | 152 | McLaughlin | Liz | NY | 19001 |
| | + | 153 | Greber | Christopher | DE | 19446 |
| | + | 154 | Flowers | Lucy | DE | 19002 |
| | + | 155 | Joranne | Scott | PA | 19422 |
| | + | 157 | Kents | Brian | NY | 19403 |
| | + | 158 | Keener | William | NJ | 19444 |
| | + | 159 | Nguyen | Ted | DC | 19401 |
| | + | 160 | Hauny | Mindy | DE | 19426 |
| | + | 161 | Burcik | Reber | DE | 19446 |
| | + | 162 | Fallows | Keith | NY | 19440 |
| | + | 163 | Rahn | Sid | PA | 19438 |
| | + | 164 | O'Mara | John | NY | 19002 |
| | + | 165 | Welsh | Irma | NY | 19473 |
| | + | 166 | Bewley | Red | NJ | 19002 |
| | + | 171 | Edgar | Ernest | PA | 19027 |
| | + | 172 | Legrow | Bill | DE | 19415 |
| | + | 173 | Davenport | David | DE | 19002 |

Record: ◄◄ ◄ 1 ► ►► ►* of 197

| First Name | Last Name | Birthday | Grade 1 | Grade 2 | Grade 3 | Final Grade |
|---|---|---|---|---|---|---|
| John | Smith | June 7, 1987 | 100 | 98 | 89 | 95.67 |
| Kathleen | Jones | October 5, 1986 | 75 | 89 | 84 | 82.67 |
| Juanita | Garcia | December 15, 1986 | 99 | 97 | 100 | 98.67 |
| Charles | Kim | February 28, 1987 | 50 | 68 | 42 | 53.33 |
| Natalie | Washington | May 4, 1987 | 85 | 87 | 79 | 83.67 |
| James | Hernandez | October 8, 1986 | 74 | 72 | 81 | 75.67 |
| William | Keener | March 21, 1987 | 32 | 54 | 67 | 51.00 |
| Seymour | Papert | April 26, 1987 | 84 | 92 | 81 | 85.67 |
| Jackie | Kennedy-Onassis | January 18, 1987 | 92 | 90 | 88 | 90.00 |
| Timothy | Smith | February 8, 1987 | 72 | 65 | 99 | 78.67 |

# Relational Databases – Constraints

| Key Constraints | Domain Constraints | Referential Integrity Constraints |
| --- | --- | --- |
| *Candidate Keys* must uniquely identify an entity. A *Candidate Key* can not have NULL values. | Every *attribute* must have a specific range of values. | A *Foreign Key* refers to a *Primary key* of a different *table*. If a *tuple* has a *Foreign Key*, that *PK* must exist. |

# Database - Keys in Entity Framework

A *key* serves as a unique identifier for each *entity* instance. Most *entities* in *EF* have one *key* which maps to the concept of a *Primary Key* in relational databases. It is possible for an *entity* to have no keys. *Entities* can also have additional *keys* (Alternate *Keys*) beyond the *Primary Key*. By default, any property named Id or <type name>Id will be automatically configured by *EF* as the *Primary Key* of an *entity*.

You can force configure any single property to be the Primary Key of an entity.

```
class Car
{
    [Key]
    public string LicensePlate { get; set; }

    public string Make { get; set; }
    public string Model { get; set; }
}
```

# Databases – Anomalies

---

If a Database has inconsistent data, it will incur **anomalies**. A DB with *anomalies* can give inconsistent data.

*There are three types of anomalies:*

- *Update anomaly* − If data items are not linked to each other properly, when one data item is updated, a few instances may get updated properly while a few others are left with old values

- *Deletion anomaly* − When a record is deleted, but linked parts of it were left undeleted because of unawareness or when deletion deletes other data unintentionally.

- *Insertion anomaly* − When data is inserted into a record that does not exist or cannot be inserted without an unrelated data.

# DataBases Competition

There are many options when it comes tochoosing your next Database.



## Database Platform Options

| PostgreSQL | | Free (Open Source) Widely used on internet Multi platform |
|---|---|---|
| MySQL MariaSQL | | Free (Open Source) Widely used on internet Multi platform. |
| MS SQL Server Express | | Free, but with some limitations Compatible with SQL Server Windows only |
| Microsoft Access | | Cost (-) Not easy to use just SQL (-) |
| SQLite | | Free (Open Source) Mainly command line (-) |