

<epam>

# Module “Data processing”

## Submodule “DB&SQL”

DB fundamentals, SQL overview

UA Resource Development Unit  
2020

# Content

- DB overview, DB types
- Relational data bases
- Relations, keys, relationships
- Normalization. Normal forms
- ACID general
- MS SQL Server и T-SQL

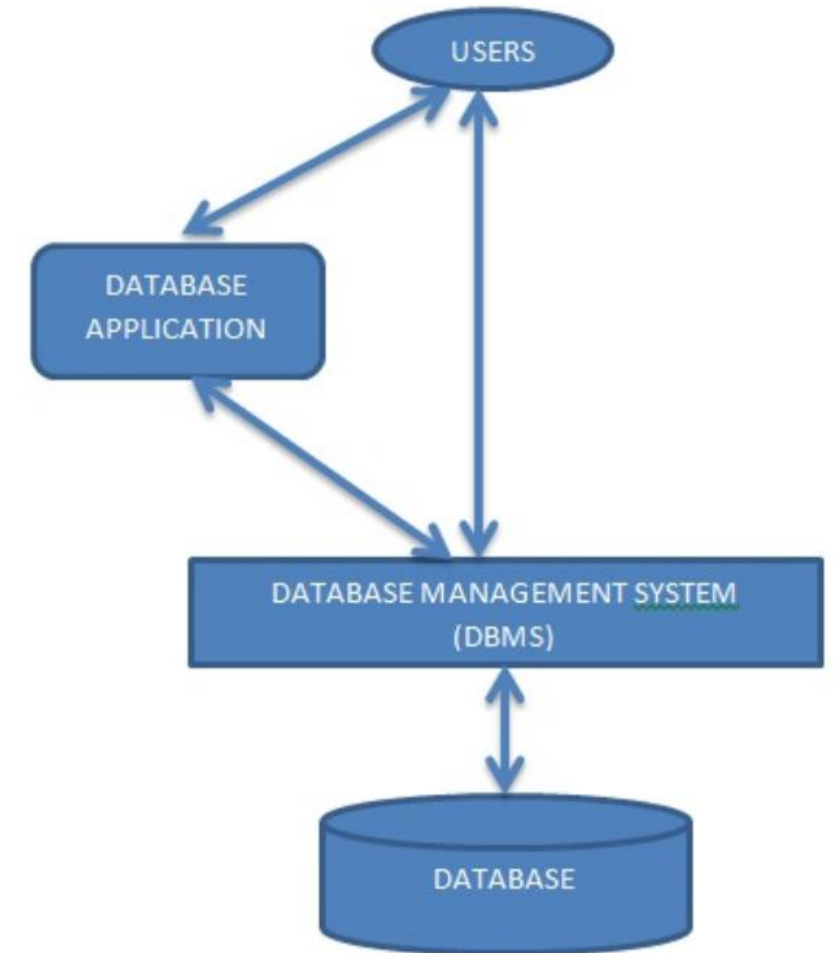
# DB overview, DB types

## DB and DBMS

### Data Base definitions:

- ✓ A database is an organized collection of data, generally stored and accessed electronically from a computer system.
- ✓ A database is a collection of information that is organized so that it can be easily accessed, managed and updated.
- ✓ A database is a data structure that stores organized information.

**Data Base Management System** - is a software package designed to define, manipulate, retrieve and manage data in a database. A DBMS generally manipulates the data itself, the data format, field names, record structure and file structure. It also defines rules to validate and manipulate this data.



# DB and DBMS

## Functions of DBMS:

- ✓ Provides data Independence
- ✓ Concurrency Control
- ✓ Provides Recovery services
- ✓ Provides Utility services
- ✓ Provides a clear and logical view of the process that manipulates data.

## Advantages of DBMS

- ✓ Segregation of application program.
- ✓ Minimal data duplicity.
- ✓ Easy retrieval of data.
- ✓ Reduced development time and maintenance need

## Disadvantages of DBMS

- ✓ Complexity
- ✓ Costly
- ✓ Large

# Relational Database

A relational database (RDB) is a collective set of multiple data sets organized by tables, records and columns. RDBs establish a well-defined relationship between database tables. Tables communicate and share information, which facilitates data searchability, organization and reporting.

RDBs use Structured Query Language (SQL)

RDB is derived from the mathematical function concept of mapping data sets and was developed by Edgar F. Codd (1970, article «A Relational Model of Data for Large Shared Data Banks».)

In the early 80's relational model became very popular. Codd published "12 Codd Rules" that described the requirements to relational databases to separate RDBs from other data sets which didn't support relational technology.

## Relational Database vs NoSQL databases

Traditional RDBs store data in highly structured tabular form, with multiple rows and columns. While these data stores are highly flexible, easy to maintain, and effective for data stored on a single server, they do not scale very well in a distributed system compared to NoSQL database alternatives.

NoSQL databases do not use the standard tabular relationships the relational databases employ. Instead, NoSQL databases allow for the querying and storage of data by a variety of other means, depending on the specific software.

There are several different types of NoSQL data model. The most popular data models used in NoSQL databases are

- ✓ **document** model,
- ✓ **graph** model,
- ✓ **key-value** model

## Relational Database vs NoSQL databases



- ✓ **Document model:** These NoSQL databases replace the familiar rows and columns structure with a document storage model. Each document is structured, frequently using the JavaScript Object Notation (JSON) model. The document data model is associated with object-oriented programming where each document is an object (use cases content management, catalog).



- ✓ **Graph model:** NoSQL databases using the graph model usually require all the data to reside on one machine which negates one of the key advantages of NoSQL databases. This class of databases uses structures like data nodes, edges and properties, making it easier to model relationships between entities in an application (use cases Recommendation engines, Fraud detection)



- ✓ **Key-value model:** In this NoSQL database model, a key is required to retrieve and update data. The key-value data model is very simple and therefore scales well. However, this simplicity and scalability come at the cost of query complexity (use cases session store, shopping cart).



# Relational data bases

## Relational Database

Briefly, the features of relational databases can be formulated as follows:

- ✓ Data is stored in tables consisting of columns ("attributes") and rows ("records", "tuples").
- ✓ There is exactly one value at the intersection of each column and line.
- ✓ Each column has a name that serves as its name, and all values in one column are of the same type.
- ✓ Database queries return results in the form of tables, which can also act as a query object.
- ✓ Rows in a relational database are unordered: the ordering is performed at the moment of generating a response to a query.
- ✓ The generally accepted language standard for working with relational databases is the SQL language.

## Advantages and disadvantages of the relational model

- ✓ simplicity and accessibility for user understanding. Only "table" is used as an information structure;
- ✓ strict design rules are based on the mathematical theory;
- ✓ complete independence of data. When the relational database is changed, appropriate changes in the applications are minimal;
- ✓ to organize queries and write application software there is no need to know the specific organization of the database in external memory.



- ✓ not always the subject area can be represented in the form of "tables";
- ✓ as a result of logical design there is a set of "tables". This leads to difficulties in understanding the data structure;
- ✓ The database takes up a relatively large amount of external memory;
- ✓ relatively low speed of data access



# Relations, keys, relationships

## Relational Model Concepts

**Tables** – In the Relational model the relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.

**Tuple** – It is a single row of a table, which contains a single record

**Attribute** - Each column in a Table. Attributes are the properties which define a relation.

**Attribute domain** – Every attribute has some pre-defined value and scope which is known as attribute domain

**Relation Schema** - A relation schema represents the name of the relation with its attributes.

**Degree** - The total number of attributes which in the relation is called the degree of the relation.

**Cardinality** - Total number of rows present in the Table.

**Relation instance** – Relation instance is a finite set of tuples in the RDBMS system. Relation instances never have duplicate tuples

# Relational Model Concepts

**Table also called Relation**

**Domain**  
Ex: NOT NULL

Supplier	Address	Product
Peter & partners	Green Street	Spoons
Peter & partners	Green Street	Forks
Roy brothers	34 avenue	Plates

**Tuple OR Row**

Total # of rows is **Cardinality**

**Column OR Attributes**

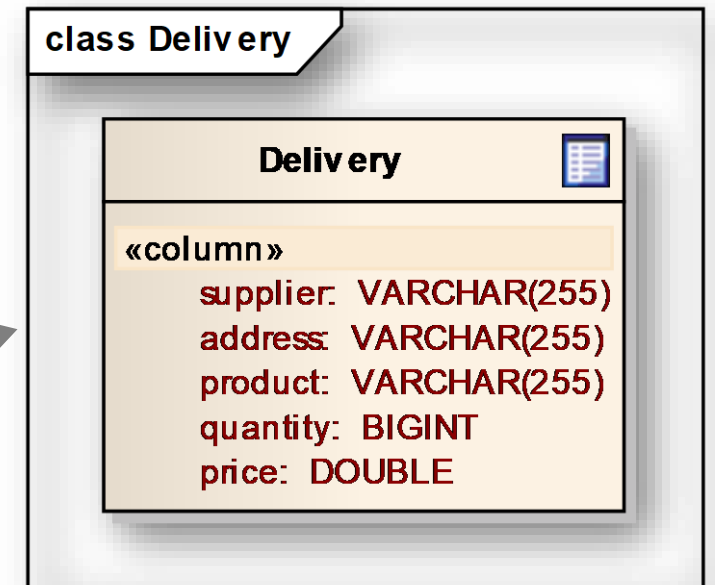
Total # of column is **Degree**

## Relational Model Concepts

Supplier	Address	Product	Quantity	Price
Peter & partners	Green Street	Spoons	34	4.6
Peter & partners	Green Street	Forks	67	5.7
Peter & partners	Green Street	Knives	45	3.9
Roy brothers	34 avenue	Plates	45	5.8
Roy brothers	34 avenue	Glasses	36	3.2

relation

UML notation



## Relational Model Concepts

SQL databases are 'relational' as they basically store relationships between data.

**Relationship** is a way of indicating the fact that two (sometimes one or more) relations are in a logical relationship with each other.

There are several types of relationships which can exist between tables

Links are organized using **keys**.



# Keys

Key is a relation attribute (or a set of attributes) that has some specific properties depending on the type of key.

Here are some reasons for using key in the DBMS system.


- ✓ Keys help you to identify any row of data in a table. In a real-world application, a table could contain thousands of records. Moreover, the records could be duplicated. Keys ensure that you can uniquely identify a table record despite these challenges.
- ✓ Allows you to establish a relationship between and identify the relation between tables
- ✓ Help you to enforce identity and integrity in the relationship.

## Primary Key

**Primary Key** - is a column or group of columns in a table that uniquely identify every row in that table.

Remember that in relational database theory, a table (relation) is an initially unordered set of records (tuples).

The single way to identify a particular record in this table is to specify a set of values for one or more fields that would be unique for that record.



StudID	Pin	FirstName	LastName	Email
1	56	Polly	Pan	<a href="mailto:pa@gmail.com">pa@gmail.com</a>
2	57	Jack	Brown	<a href="mailto:br@@gmail.com">br@@gmail.com</a>
3	58	Katy	Green	<a href="mailto:gr@gmail.com">gr@gmail.com</a>

# Primary Key

The primary key serves to:

- ✓ identification of the table row;
- ✓ organizing relationships between tables.

## **Rules for defining Primary key:**

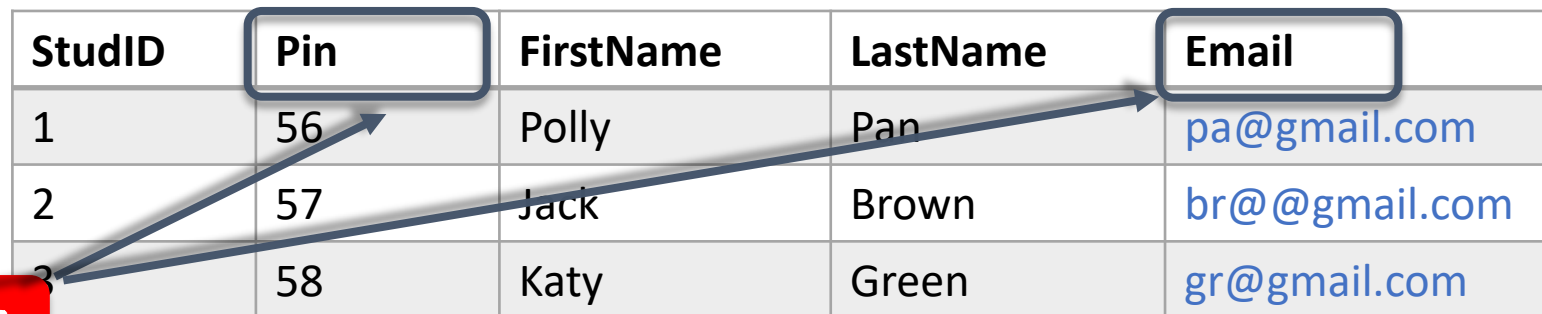
- ✓ Two rows can't have the same primary key value
- ✓ It must for every row to have a primary key value.
- ✓ The primary key field cannot be null.
- ✓ The value in a primary key column can never be modified or updated if any foreign key refers to that primary key.

## Alternate key

**Alternate keys** is a column or group of columns in a table that uniquely identify every row in that table. A table can have multiple choices for a primary key but only one can be set as the primary key. All the keys which are not primary key are called an Alternate Key.

In this table, StudID, Pin, Email are qualified to become a primary key. But since StudID is the primary key, Pin, Email becomes the alternative key.

StudID	Pin	FirstName	LastName	Email
1	56	Polly	Pan	pa@gmail.com
2	57	Jack	Brown	br@@gmail.com
3	58	Katy	Green	gr@gmail.com



alternate keys

## Candidate Key

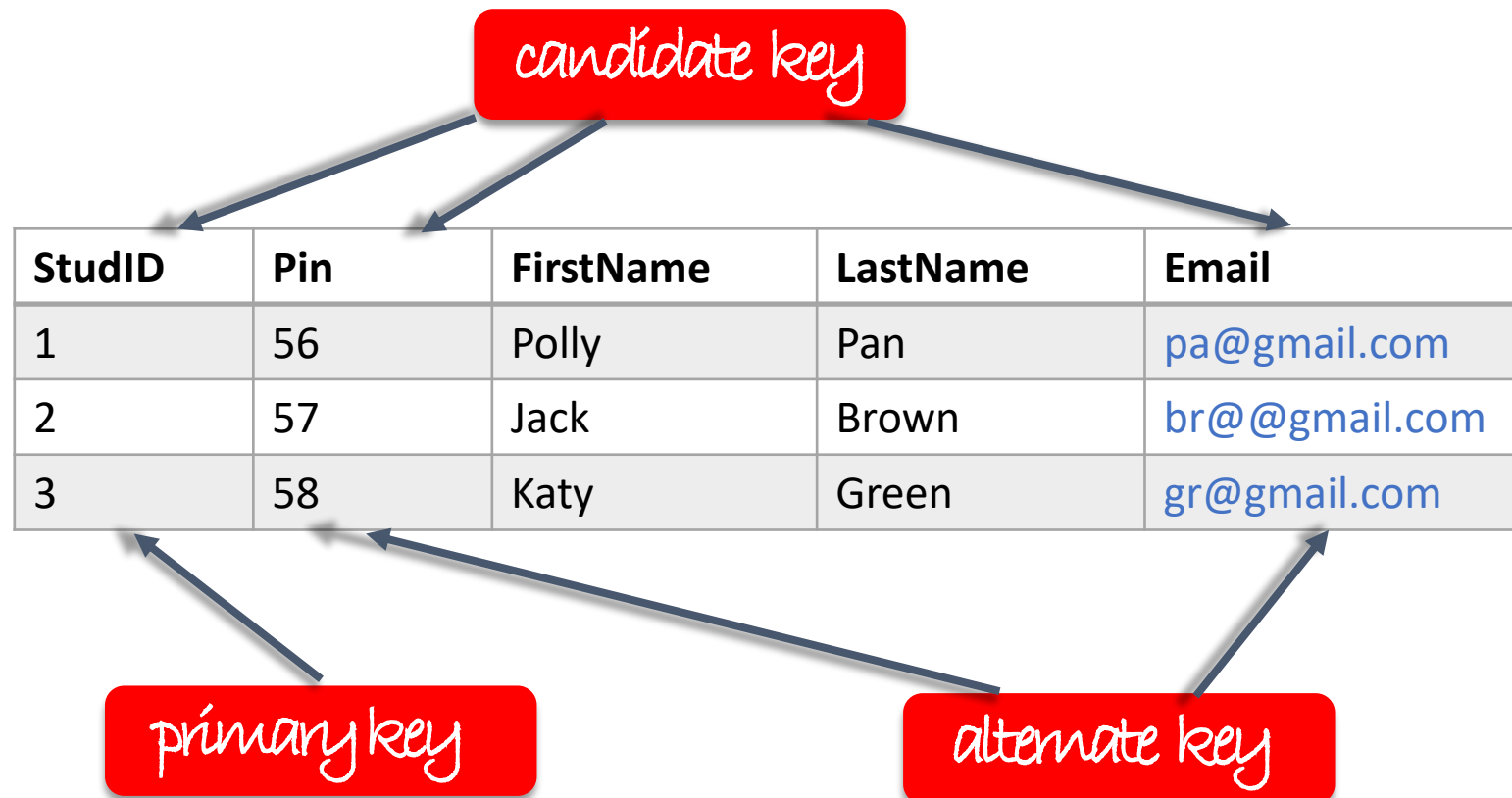
**Candidate key** is a set of attributes that uniquely identify tuples in a table. Candidate Key is a super key with no repeated attributes. The Primary key should be selected from the candidate keys. Every table must have at least a single candidate key. A table can have multiple candidate keys but only a single primary key.

### **Properties of Candidate key:**

- It must contain unique values
- Candidate key may have multiple attributes
- Must not contain null values
- It should contain minimum fields to ensure uniqueness
- Uniquely identify each record in a table

Example: in the given table Stud ID, Pin and email are candidate keys which help us to uniquely identify the student record in the table


## Primary, Alternative and Candidate Keys



## Simple and Composite Keys

**Simple key** consists of single column that uniquely identifies rows in a table.

**Composite key** is a combination of two or more columns that uniquely identify rows in a table. The combination of columns guarantees uniqueness, though individually uniqueness is not guaranteed. Hence, they are combined to uniquely identify records in a table.




OrderID	ProductID	ProductName	Quantity
0067	JP23456t	Spoons	5
0067	SD2343rt	Forks	30
0067	CV5657c	Knives	23
0066	KH46669q	Plates	77
0066	UI475585z	Glasses	45
0065	SD2343rt	Forks	7

## Natural Primary Key

The primary key can consist of table information fields (that is, fields containing useful information about the described objects). Such primary key is called a **natural key**.

In practice, the use of natural keys runs into certain difficulties:

- ✓ low efficiency (size, need to duplicate in case of migration);
- ✓ cascading changes;
- ✓ inconsistency with reality (the value of the field selected as the primary key can be specified later).



OrderID	ProductID	ProductName	Quantity
0067	JP23456t	Spoons	5
0067	SD2343rt	Forks	30
0067	CV5657c	Knives	23



## Surrogate primary key

**Surrogate keys** is an artificial key which aims to uniquely identify each record is called a surrogate key. This kind of partial key in dbms is unique because it is created when you don't have any natural primary key. They do not lend any meaning to the data in the table. Surrogate key is usually an integer. A surrogate key is a value generated right before the record is inserted into a table.



ID	FName	LName	Start	End
1	Polly	Lawrence	12:00	13:00
2	Morgan	Milles	14:00	15:00
3	Amber	Tran	10:00	12:00
4	Isobel	Cooper	18:00	19:00

Above, given example, shown shift timings of the different employee. In this example, a surrogate key is needed to uniquely identify each employee.

## Natural vs Surrogate keys

### Problems with PK "Passport":

- ✓ Big size.
- ✓ Requires for cascading changes.
- ✓ Impossible to insert data in case of information absence



### Advantages of PK "ID":

- ✓ Small size.
- ✓ Not requires for cascading changes.
- ✓ You can insert data even if some information is missing



## Foreign Key

**Foreign** is a column that creates a relationship between two tables. The purpose of Foreign keys is to maintain data integrity and allow navigation between two different instances of an entity. It acts as a cross-reference between two tables as it references the primary key of another table.

DeptCode	DeptName
001	Science
002	English
005	Computer

Parent table Departments

Teacher ID	DeptCode	Fname	Lname
B002	002	David	Warner
B017	002	Sara	Joseph
B009	001	Mike	Brunton

Child table Teachers

## Primary vs Foreign keys

### Primary:

- ✓ Helps you to uniquely identify a record in the table.
- ✓ Primary Key never accept null values.
- ✓ Primary key is a clustered index and data in the DBMS table are physically organized in the sequence of the clustered index.
- ✓ You can have the single Primary key in a table.

### Foreign:

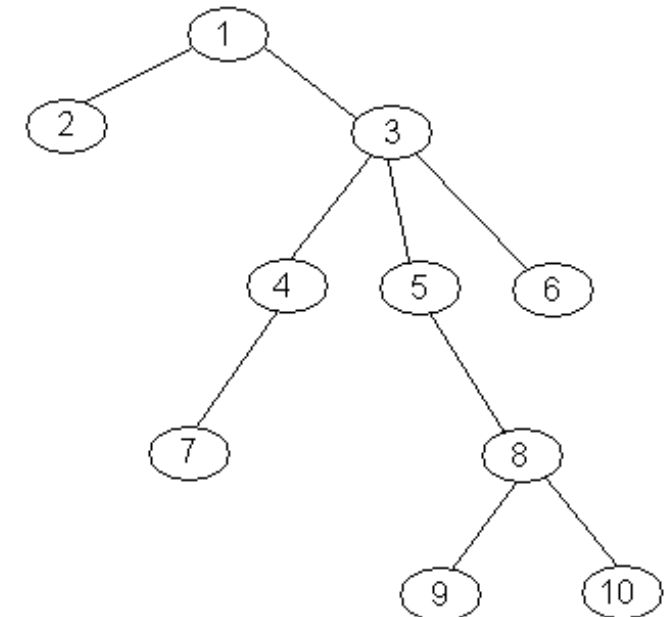
- ✓ It is a field in the table that is the primary key of another table.
- ✓ A foreign key may accept multiple null values
- ✓ A foreign key cannot automatically create an index, clustered or non-clustered. However, you can manually create an index on the foreign key
- ✓ You can have multiple foreign keys in a table

## Recursive Foreign Key

**Recursive foreign key** is a **foreign key** which refers back to the primary **key** in the same table where child and parent table are same.

Employee ID	Manager Id	Fname	Lname
1	0	David	Warner
2	1	Sara	Joseph
3	1	Mike	Brunton

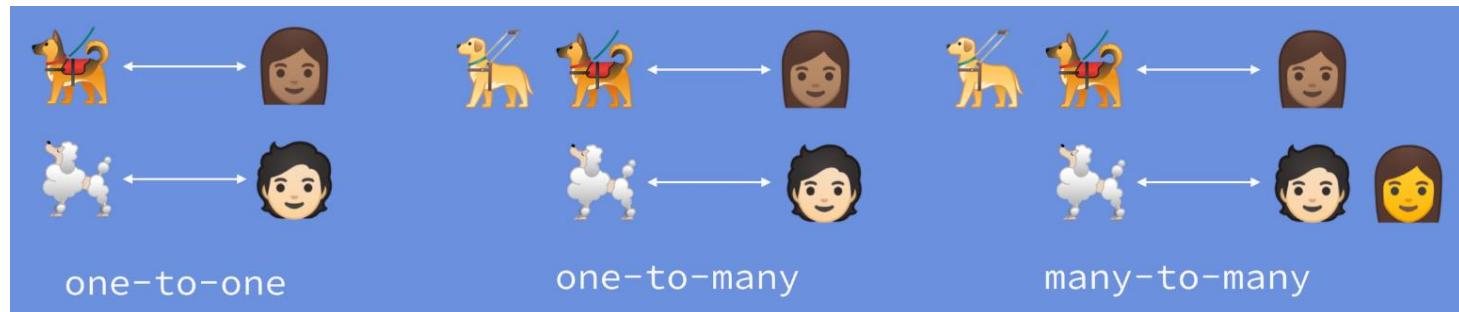
Parent and child table Employees



## Relationships

The relationship type depends on how the linked columns are defined.  
There are three main types of communication:

- ✓ one to one
- ✓ one to many
- ✓ many to many



## Relationships

**One to One:** One table record relates to another record in another table.

EmployeeID	FName	LName
1	Peter	Pan
2	Jack	Brown
3	Katy	Green

Parent table Employees

EmployeeID	Address	InsuranceCard
1	5 av.	3456789
2	Freedom sq.	2349674
3	Times sq.	3435286

Child table Profiles

splitting a table with many columns

isolation of the table part for security reasons

storing short-term data that can be easily deleted

## Relationships

**One to Many:** One table record relates to many records in another table.

Room	Floor	Name
101	1	Cuba
201	2	Bali
202	3	Panama

Parent table Rooms

EmployeeID	Room	Place
1	101	34
2	201	56
3	101	67

Child table Space

Case 1 - FK NOT  
NULL

Case 2 - FK CAN BE NULL

Case 3 - FK NOT  
NULL, by default



## Relationships

**Many to Many:** More than one table record relates to more than one record in another table

1 **table Rooms**

EmployeeID	FName	LName
1	Peter	Pan
2	Jack	Brown
3	Katy	Green

EmployeeID	ProjectId	Role
1	1	Development
2	2	Analyst
1	2	Testing
3	1	Testing

ProjectID	Description	Customer
1	E-commerce	Spark Inc.
2	CRM	ASW style
3	Web portal	Liga school

**table Projects**

Additional table is required

ID can be added to additional table

System properties can be added to additional table

# Normalization.

## Normal forms

# Normalization

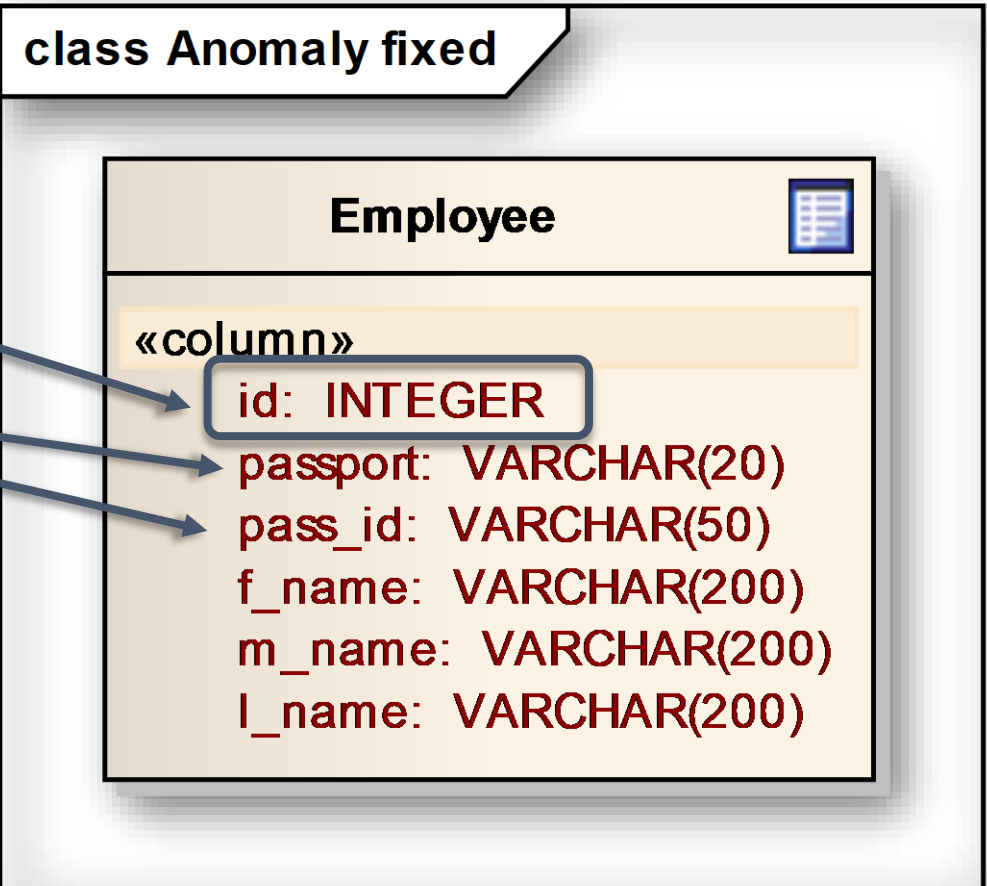
**NORMALIZATION** is a database design technique that reduces data redundancy and eliminates undesirable characteristics like Insertion, Update and Deletion Anomalies. Normalization rules divides larger tables into smaller tables and links them using relationships. The purpose of Normalization in SQL is to eliminate redundant (repetitive) data and ensure data is stored logically.

## Normalization requirements

Relationship primary keys should be minimal

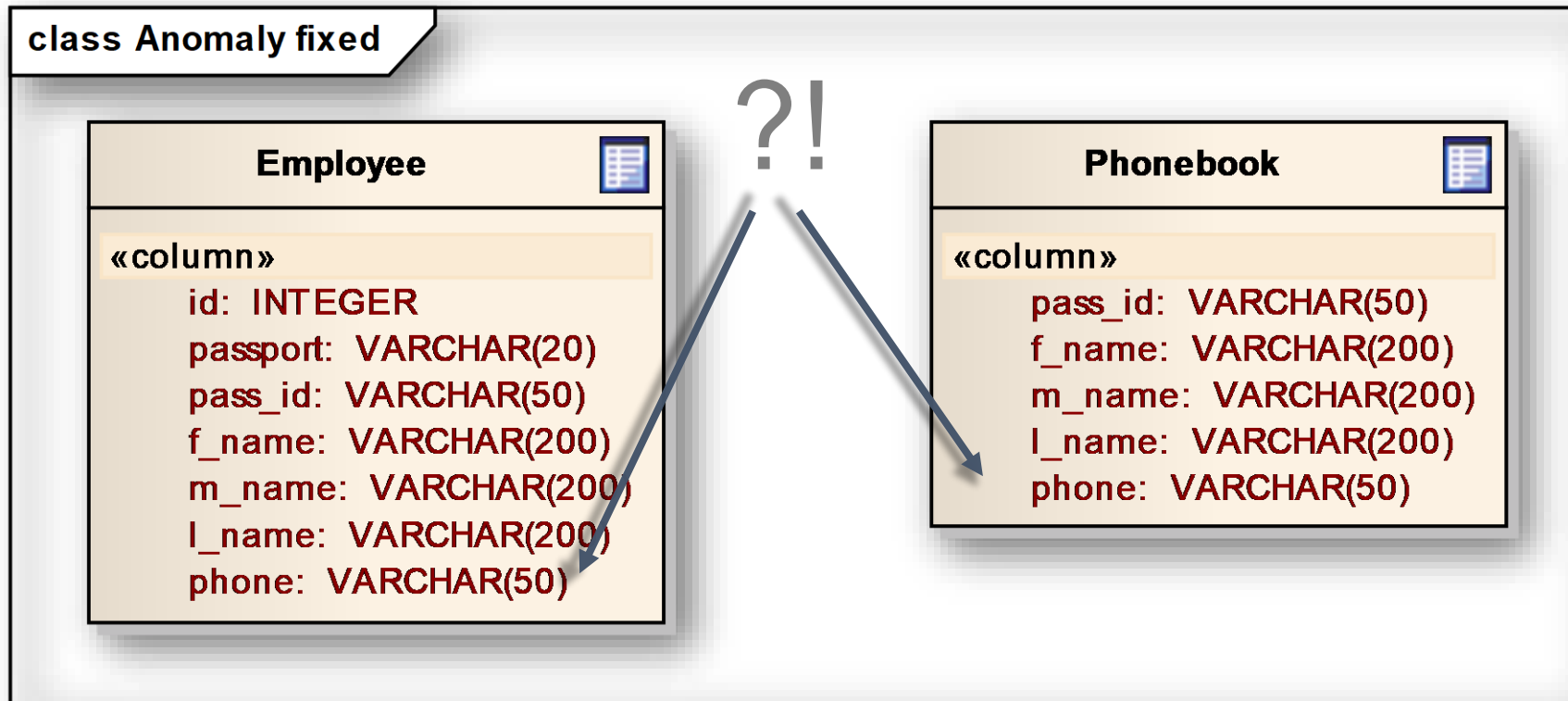
This requirement is perfectly met with the introduction of surrogate PKs

!  
?



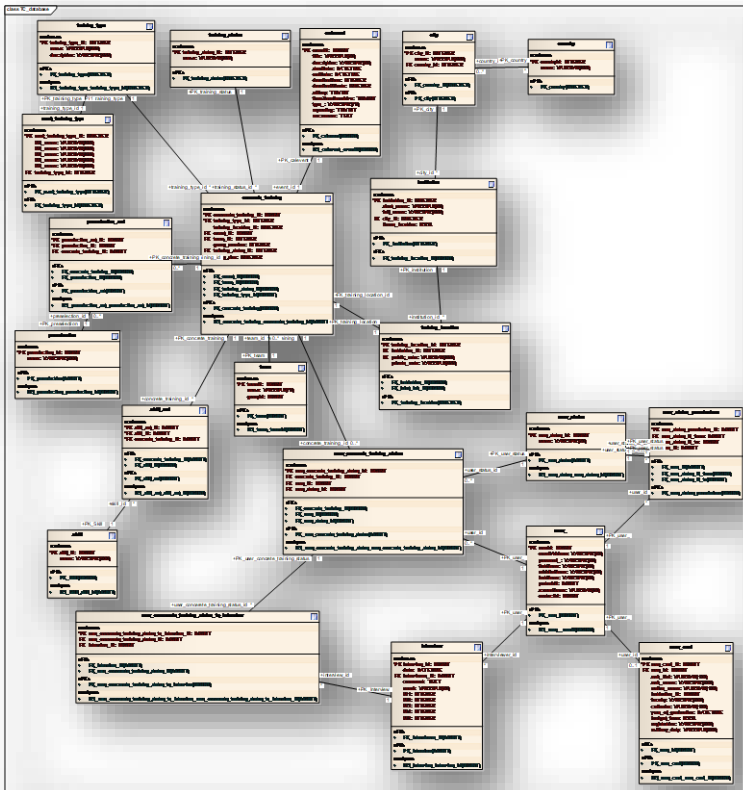
## Normalization requirements

The database model should minimize or eliminate data redundancy whenever possible



# Normalization requirements

The database model should provide the required performance of operations



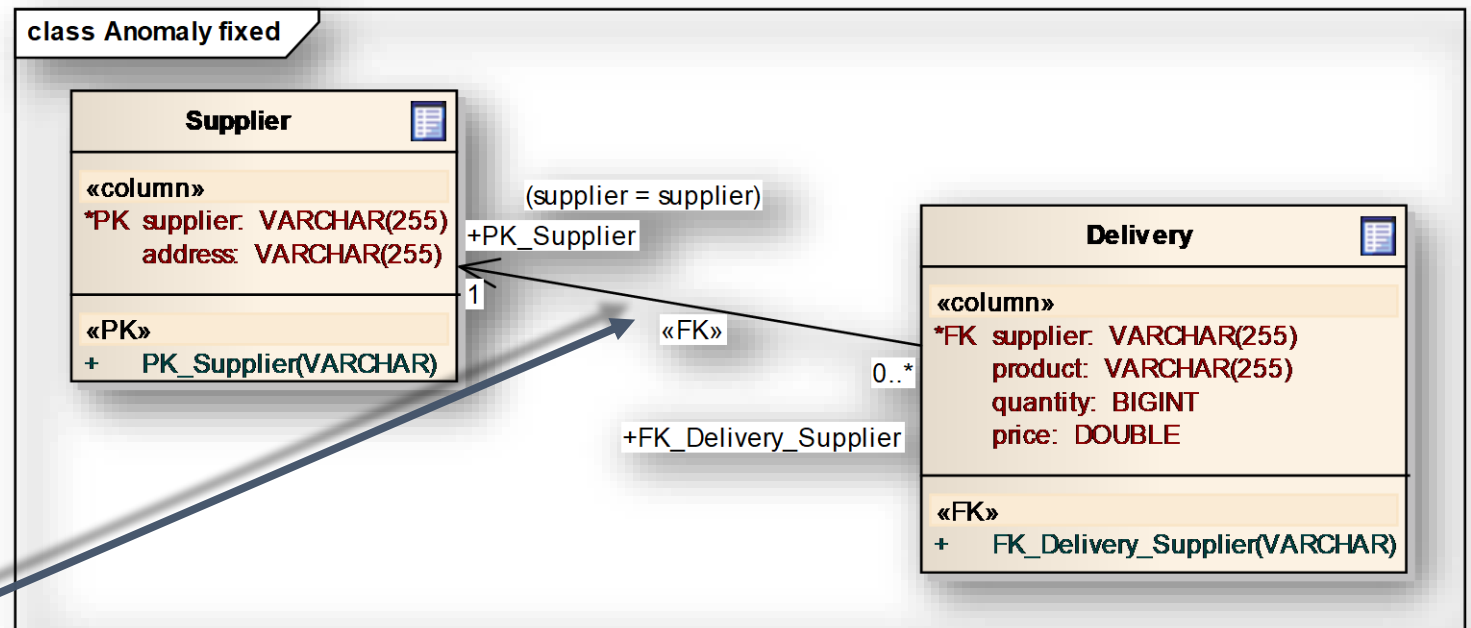
```
SELECT `news_keywords`.`nk_url`,
`news_keywords`.`nk_name`,
count(*) as 'q' from
`news_keywords` RIGHT JOIN
`n_m2m_nk` ON
`news_keywords`.`nk_uid`=`n_m2m_nk`.`nk_uid` group by
`n_m2m_nk`.`nk_uid` order by `q`
desc, `news_keywords`.`nk_name`
asc limit 100
```

?...

## Normalization requirements

The database model should minimize data inconsistency in all data operations

The relationships must be established EXPLICITLY!



## Normalization requirements

The database model must be adaptable if changes are required

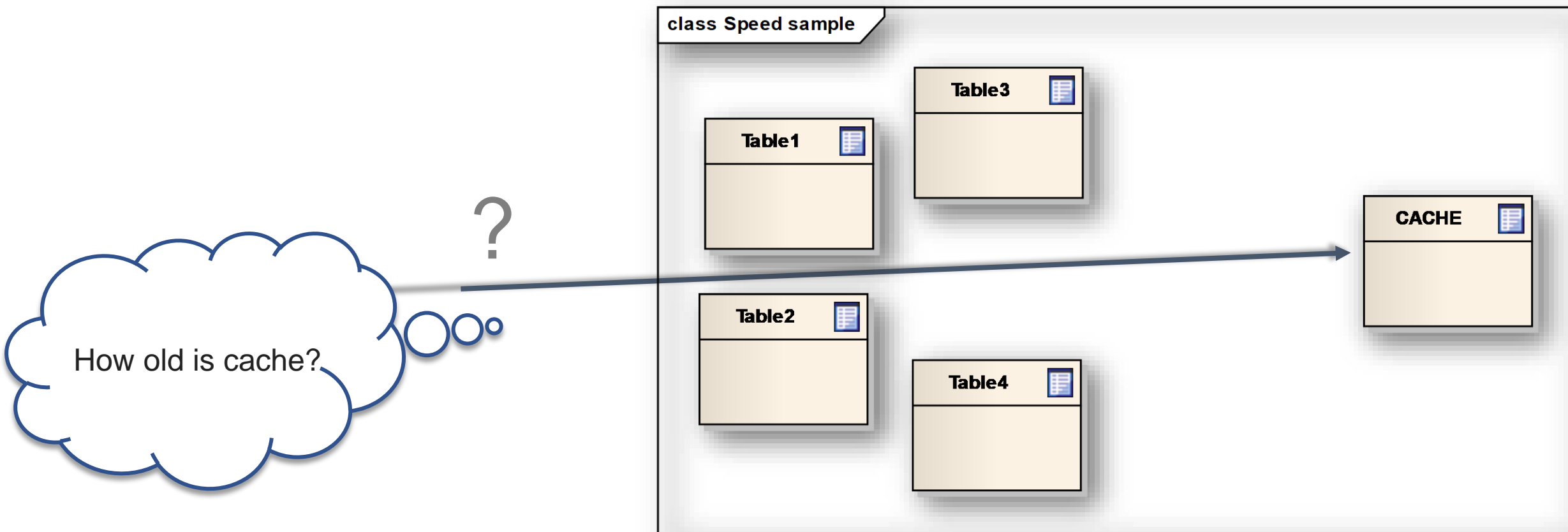
This is achieved by:

- ✓ Mnemonic names.
- ✓ Comments.
- ✓ Documentation.
- ✓ Schemes in generally accepted notation.
- ✓ No stupid restrictions



## Normalization requirements

Everywhen the database must contain an actual data set



## Anomalies in Data Bases

Anomaly is contradiction between the domain model and the data model supported by the means of a specific DBMS.

This problems can occur in poorly planned, unnormalized databases where all data is stored in one table (a flat-file database)

Types of anomalies are

- ✓ Insert
- ✓ Update
- ✓ Delete



## Insert Anomaly

An Insert Anomaly occurs when certain attributes cannot be inserted into the database without the presence of other attributes

CourseNo	Tutor	Room	RoomSize	Media
456	Tran	K567	20	3
234	Cooper	J234	50	7
186	Tran	L344	100	5
455	Jackson	S345	30	2
212	Carey	K567	20	3

e.g. we have built a new room (e.g. J234) but it has not yet been timetabled for any courses or members of staff.

## Delete Anomaly

A Delete Anomaly exists when certain attributes are lost because of the deletion of other attributes

CourseNo	Tutor	Room	RoomSize	Media
456	Tran	K567	20	3
234	Cooper	J235	50	7
186	Tran	L344	100	5
455	Jackson	S345	30	2
212	Carey	K567	20	3

e.g. if we remove the entity, course\_no:234 from the above table, the details of room J235 get deleted. Which implies the corresponding course will also get deleted.

## Update Anomaly

An Update Anomaly exists when one or more instances of duplicated data is updated, but not at all.

CourseNo	Tutor	Room	RoomSize	Media
456	Tran	K567	20	3
234	Cooper	J234	50	7
186	Tran	L344	100	5
455	Jackson	S345	30	2
212	Carey	K567	20	3

e.g. Room K567 has been improved, it is now of Room size=20. For updating a single entity, we have to update all other columns where Room=K567.

# Database Normal Forms

Here is a list of Normal Forms

- ✓ 1NF (First Normal Form)
- ✓ 2NF (Second Normal Form)
- ✓ 3NF (Third Normal Form)
- ✓ BCNF (Boyce-Codd Normal Form)
- ✓ 4NF (Fourth Normal Form)
- ✓ 5NF (Fifth Normal Form)
- ✓ 6NF (Sixth Normal Form)

**However, in most practical applications, normalization achieves its best in 3<sup>rd</sup> Normal Form.**

## Database Normal Forms

Assume, a video library maintains a database of movies rented out. Without any normalization, all information is stored in one table as shown below.

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.
Robert Phil	3 <sup>rd</sup> Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.
Robert Phil	5 <sup>th</sup> Avenue	Clash of the Titans	Mr.

**Movies Rented column has multiple values.** Now let's move into 1st Normal Forms

# Database Normal Forms

## 1NF (First Normal Form) Rules

- ✓ Each table cell should contain a single value (An attribute will be considered atomic if there is no operation in the domain that would require retrieving part of the attribute).
- ✓ Each record needs to be unique





## Database Normal Forms


### table in 1NF

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3 <sup>rd</sup> Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3 <sup>rd</sup> Street 34	Daddy's Little Girls	Mr.
Robert Phil	5 <sup>th</sup> Avenue	Clash of the Titans	Mr.

picture source <https://www.guru99.com/database-normalization.html>

## Database Normal Forms

In our database, we have two people with the same name Robert Phil, but they live in different places.



Robert Phil	3 <sup>rd</sup> Street 34	Daddy's Little Girls	Mr.
Robert Phil	5 <sup>th</sup> Avenue	Clash of the Titans	Mr.

*Names are common. Hence you need name as well Address to uniquely identify a record.*

Hence, we require both Full Name and Address to identify a record uniquely. That is a composite key.  
Let's move into second normal form 2NF

picture source <https://www.guru99.com/database-normalization.html>

# Database Normal Forms

## 2NF (Second Normal Form) Rules

- Rule 1- Be in 1NF
- Rule 2- Single Column Primary Key (in this case 2NF completed automatically)
- Or  
every attribute depends upon the single primary key and every attribute depends upon the whole key (composite key)

# Database Normal Forms

## tables in 2NF

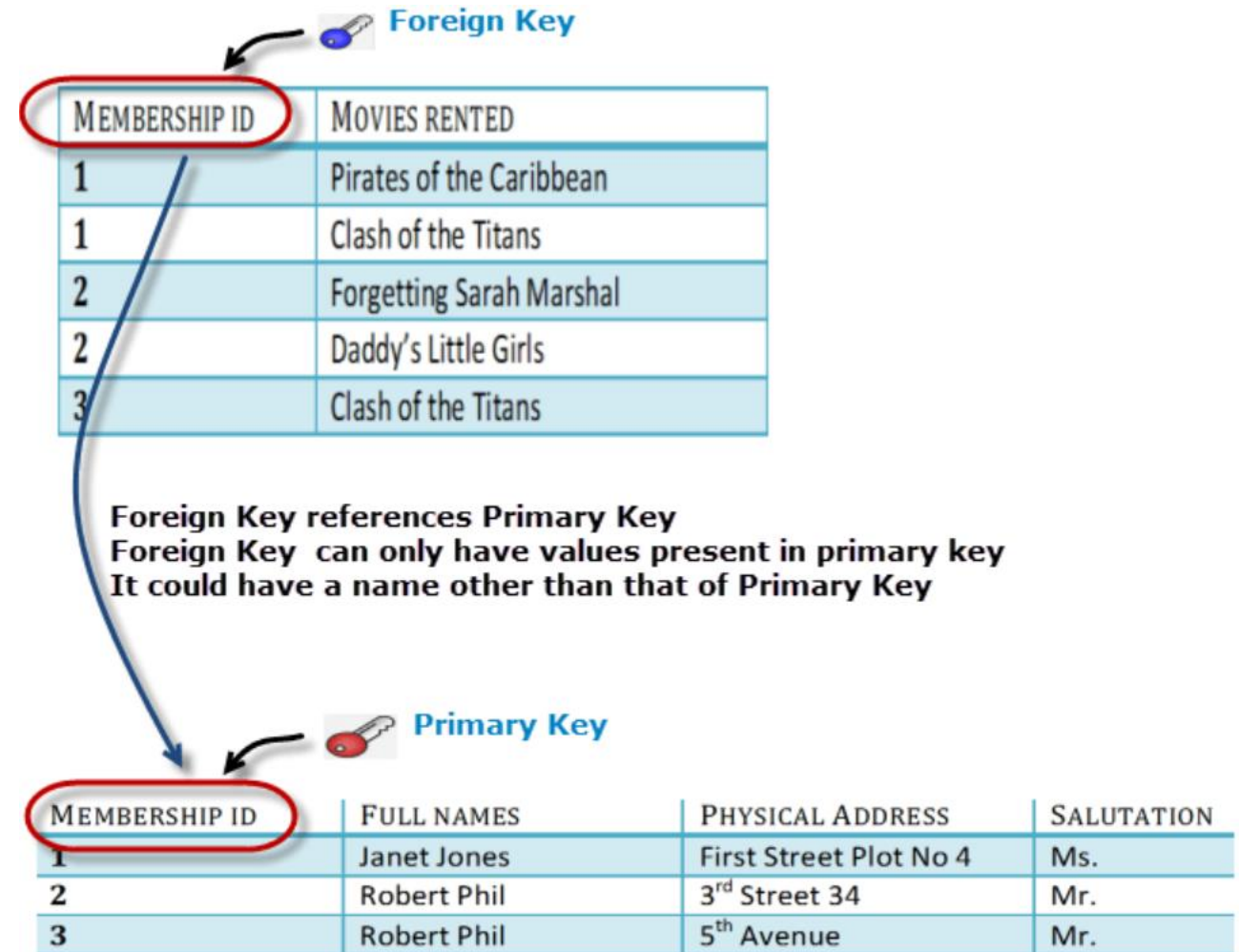
MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 <sup>rd</sup> Street 34	Mr.
3	Robert Phil	5 <sup>th</sup> Avenue	Mr.

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

# Database Normal Forms

## divide table – create foreign key!

- ✓ A foreign key can have a different name from its primary key
- ✓ It ensures rows in one table have corresponding rows in another
- ✓ Unlike the Primary key, they do not have to be unique. Most often they aren't
- ✓ Foreign keys can be null even though primary keys can not



picture source <https://www.guru99.com/database-normalization.html>

# Database Normal Forms

Why do you need a foreign key?

Insert a record in Table 2 where Member ID = 101

MEMBERSHIP ID	MOVIES RENTED
101	Mission Impossible

But Membership ID 101 is not present in Table 1

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 <sup>rd</sup> Street 34	Mr.
3	Robert Phil	5 <sup>th</sup> Avenue	Mr.

Database will throw an **ERROR**. This helps in referential integrity

picture source <https://www.guru99.com/database-normalization.html>

## Database Normal Forms

### What are transitive functional dependencies?

A transitive functional dependency is when changing a non-key column, might cause any of the other non-key columns to change

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3 <sup>rd</sup> Street 34	Mr.
3	Robert Phil	5 <sup>th</sup> Avenue	Mr.

*Change in Name* (circled around 'Robert Phil' in row 3, column 2)

*May Change Salutation* (arrow pointing from the circled name to the 'Mr.' in row 3, column 4)

Changing the non-key column Full Name may change non-key column Salutation Now let's move into 3NF

picture source <https://www.guru99.com/database-normalization.html>

# Database Normal Forms

## 3NF (Third Normal Form) Rules

- ✓ Rule 1- Be in 2NF
- ✓ Rule 2- Has no transitive functional dependencies

To move 2NF table into 3NF, often it is required to again divide table.



## Database Normal Forms

### tables in 3NF

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3 <sup>rd</sup> Street 34	1
3	Robert Phil	5 <sup>th</sup> Avenue	1

SALUTATION ID	SALUTATION
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

# Database Normal Forms

## SUMMARY

- ✓ Database designing is critical to the successful implementation of a database management system that meets the data requirements of an enterprise system.
- ✓ Normalization in DBMS helps produce database systems that are cost-effective and have better security models.
- ✓ Functional dependencies are a very important component of the normalize data process
- ✓ Most database systems are normalized database up to the third normal forms.
- ✓ A primary key uniquely identifies are record in a Table and cannot be null
- ✓ A foreign key helps connect table and references a primary key

# Database Normal Forms

Let's train

**What's  
wrong?**

Firm	Model
BMW	M5, X5M, M1
Nissan	GT-R



**OK!**

Firm	Model
BMW	M5
Nissan	GT-R
BMW	X5M
BMW	M1

# Database Normal Forms

Let's train

**What's  
wrong?**

Firm	Model	Price	Discount
BMW	M5	55	5%
Nissan	GT-R	50	10%
BMW	X5M	60	5%
BMW	M1	25	5%



**OK!**

Firm	Model	Price
BMW	M5	55
Nissan	GT-R	50
BMW	X5M	60
BMW	M1	25

Firm	Discount
BMW	5%
Nissan	10%

# Database Normal Forms

Let's train

**What's  
wrong?**

Model	Shop	Tel
BMW	Longer	55-56
Nissan	Road	50-44
Audi	Road	50-44



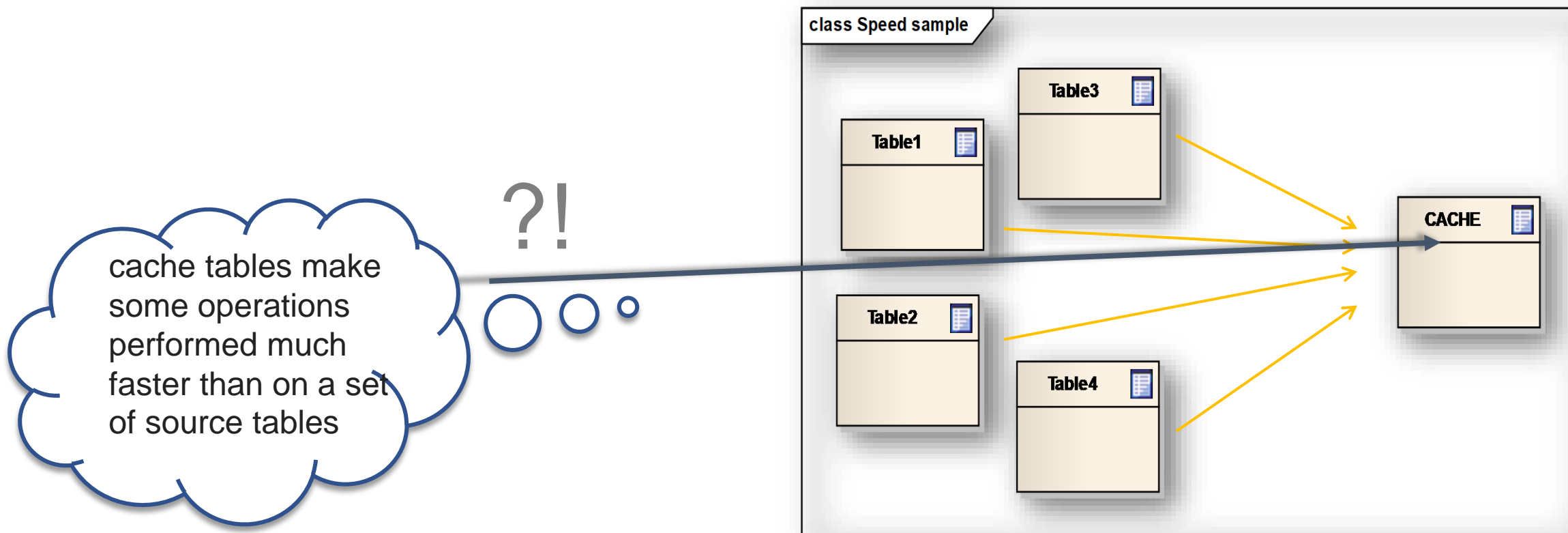
**OK!**

Model	Shop
BMW	Longer
Nissan	Road
Audi	Road

Shop	Tel
Longer	55-56
Road	50-44

# Database Normal Forms

Denormalization - the process of making relationships to a state that violates certain normal forms



# ACID

**ACID** is a set of properties of DB operations intended to guarantee validity even in the event of errors, power failures, etc. In the context of DBs, a sequence of database operations that satisfies the ACID properties (which can be perceived as a single logical operation on the data) is called a **transaction**.



# ACID

ACID is an acronym that stands for

- ✓ Atomicity
- ✓ Consistency
- ✓ Isolation
- ✓ Durability

**A** - Transferring funds from one account to another: 2 update in the table of accounts and 1 insert in the table of transfers. In this case, operations constitute a unit of work: either all or nothing required

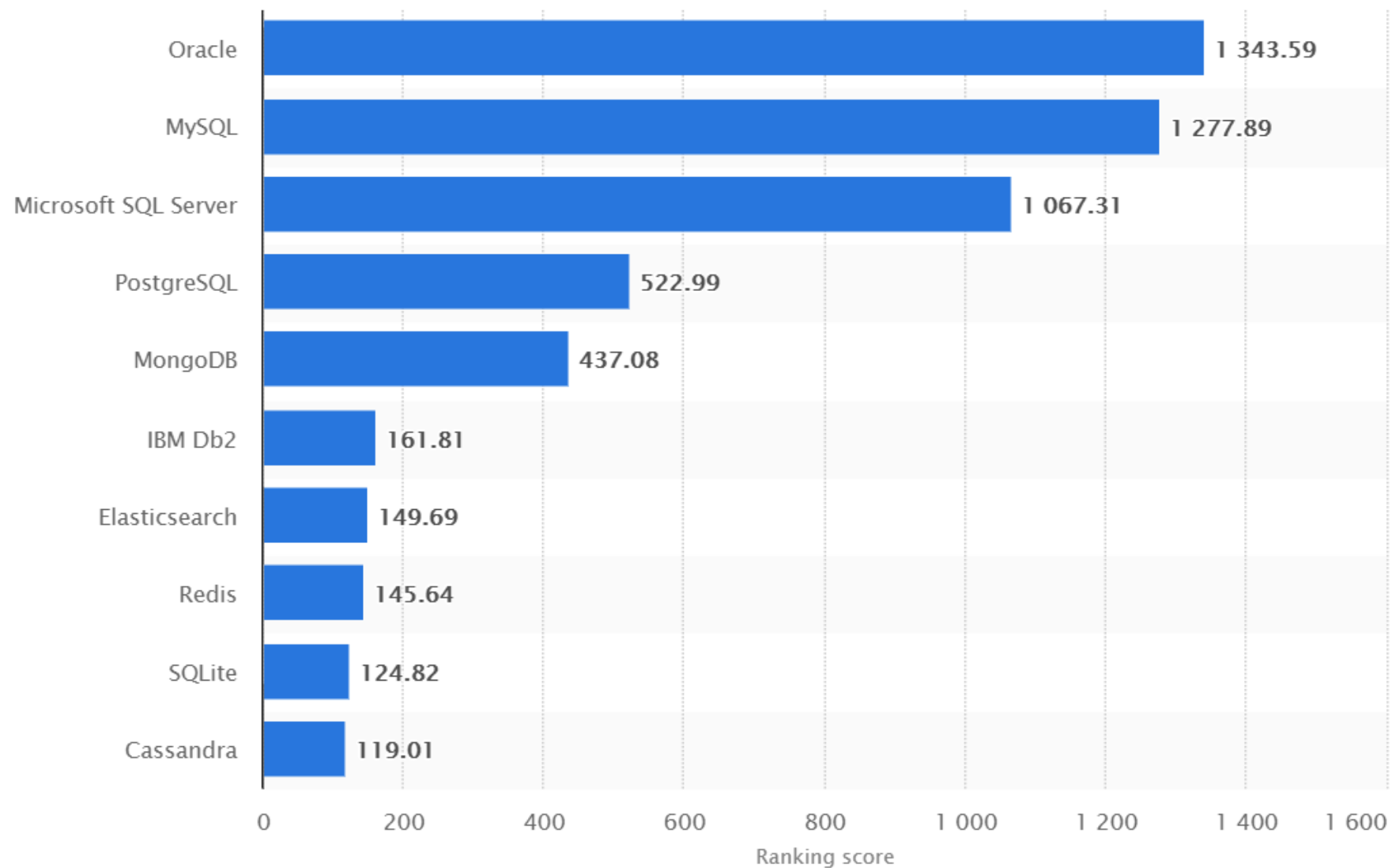
**C** - Changes do not contradict database restrictions: PK, FK, check

**I** - The result should be visible only after performing all operations

**D** - After the operation is completed, the data is available and lost



## Most Popular DBMSs (<https://www.statista.com/> June 2020)



# SQL

SQL is a standard language for storing, manipulating and retrieving databases

SQL stands for Structured Query Language

SQL lets you access and manipulate databases

SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987

SQL is more or less standardized, and used by almost all relational database management systems: SQL Server, Oracle, MySQL, PostgreSQL, DB2, Informix, etc.

PL/SQL is a proprietary procedural language used by Oracle

PL/pgSQL is a procedural language used by PostgreSQL

**TSQL is a proprietary procedural language used by Microsoft in SQL Server.**

# T-SQL

T-SQL (Transact-SQL) is a set of programming extensions from Sybase and Microsoft that add several features to the Structured Query Language (SQL), including flow statements if, while, transaction control, exception and error handling, row processing and declared variables.

## Difference between T-SQL and SQL

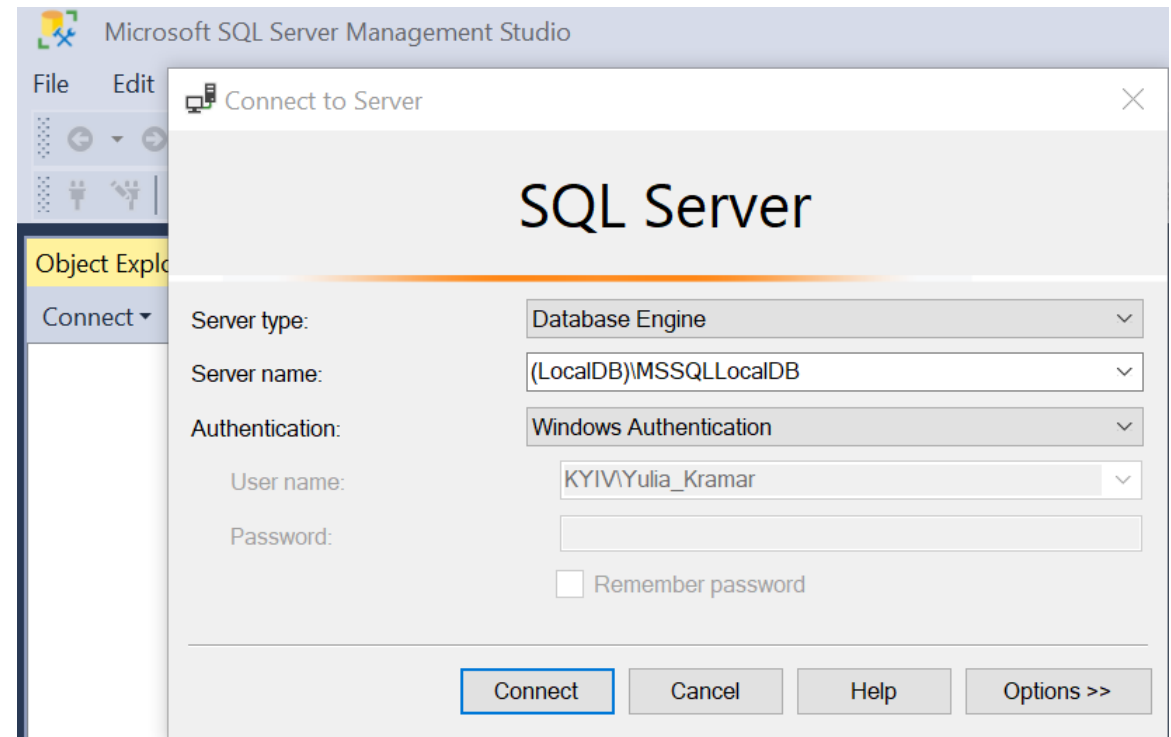
- ✓ While T-SQL is an extension to SQL, SQL is a programming language.
- ✓ T-SQL contains procedural programming and local variable, while SQL does not.
- ✓ T-SQL is proprietary, while SQL is an open format.

The most popular T-SQL statements are the stored procedure and user functions, which are a compiled and stored T-SQL code.

## Tools that use T-SQL

Some of the Microsoft tools that issue T-SQL commands are:

- [SQL Server Management Studio \(SSMS\)](#)
- [SQL Server Data Tools \(SSDT\)](#)
- [sqlcmd](#)
- [Azure Data Studio](#)



## T-SQL categories

- ✓ Data Manipulation Language (DML)

**INSERT INTO** (*Table*) **VALUES** (*list of values*)

- ✓ Data Query Language (DQL)

**SELECT** (*list of columns*) **FROM** (*table*)

- ✓ Data Definition Language (DDL)

**CREATE** (*object*) (*name*)

- ✓ Transaction Control Language (TCL)

**COMMIT TRANSACTION** (*name*)

- ✓ Data Control Language (TCL)

**GRANT CONTROL** (*object*) **TO** (*user*)

## T-SQL

BD object	Description
Tables	Tables are database objects that contain all the data in a database.
Views	A view is a virtual table whose contents are defined by a query. Like a table, a view consists of a set of named columns and rows of data.
Indexes	Speed up data access operations, indexes are generated based on one or more fields
Triggers	Triggers is a special type of stored procedure that automatically takes effect when a DML event takes place that affects the table.
Stored procedures	A stored procedure in SQL Server is a group of one or more Transact-SQL statements
User-defined functions	User-defined functions are routines that accept parameters, perform an action, such as a complex calculation, and return the result of that action as a value.
Constraints	Constraints are used to specify rules for the data in a table. Constraints are used to limit the type of data that can go into a table.

# Content

- <https://www.trustradius.com/nosql-databases>
- <https://aws.amazon.com/nosql/>
- <https://www.guru99.com/relational-data-model-dbms.html>
- <https://www.guru99.com/dbms-keys.html>
- <https://www.slideshare.net/BaabtraMentoringPartner/anomalies-in-database>
- <https://www.guru99.com/database-normalization.html>

# Q&A



DRIVEN



CANDID



CREATIVE



ORIGINAL



INTELLIGENT



EXPERT

UA .NET Online LAB