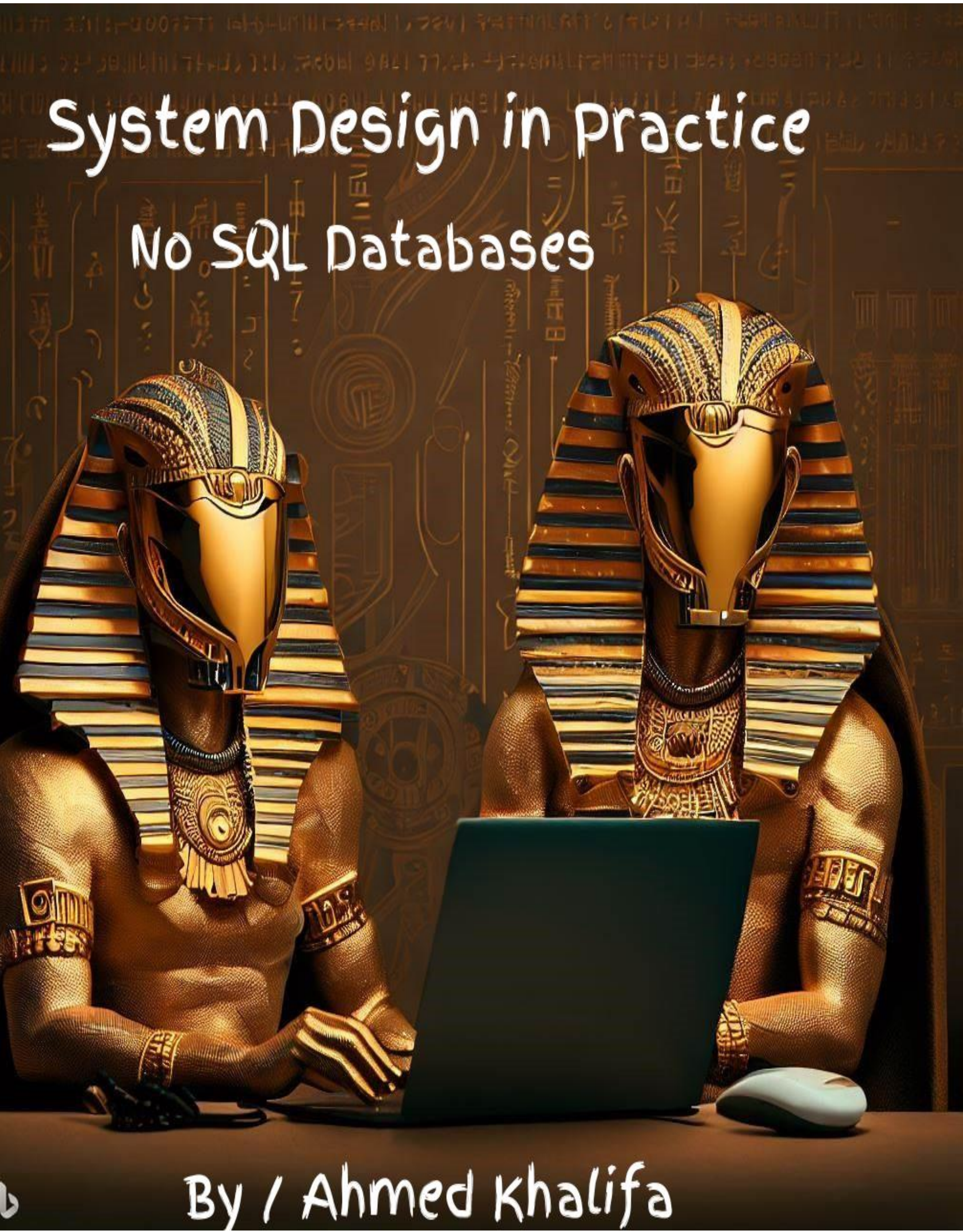
System Design in Practice

No SQL Databases



By / Ahmed Khalifa

Index

Technology agnostic :

Wrong database selection :

One of the major mistakes I see in the software industry is choosing the database based on the technology stack for example:

- If the developers team use .Net stack , the database will be automatically SQL Server.
- If the developers team use MEAN stack , the database will be automatically MongoDB.

Good database selection :

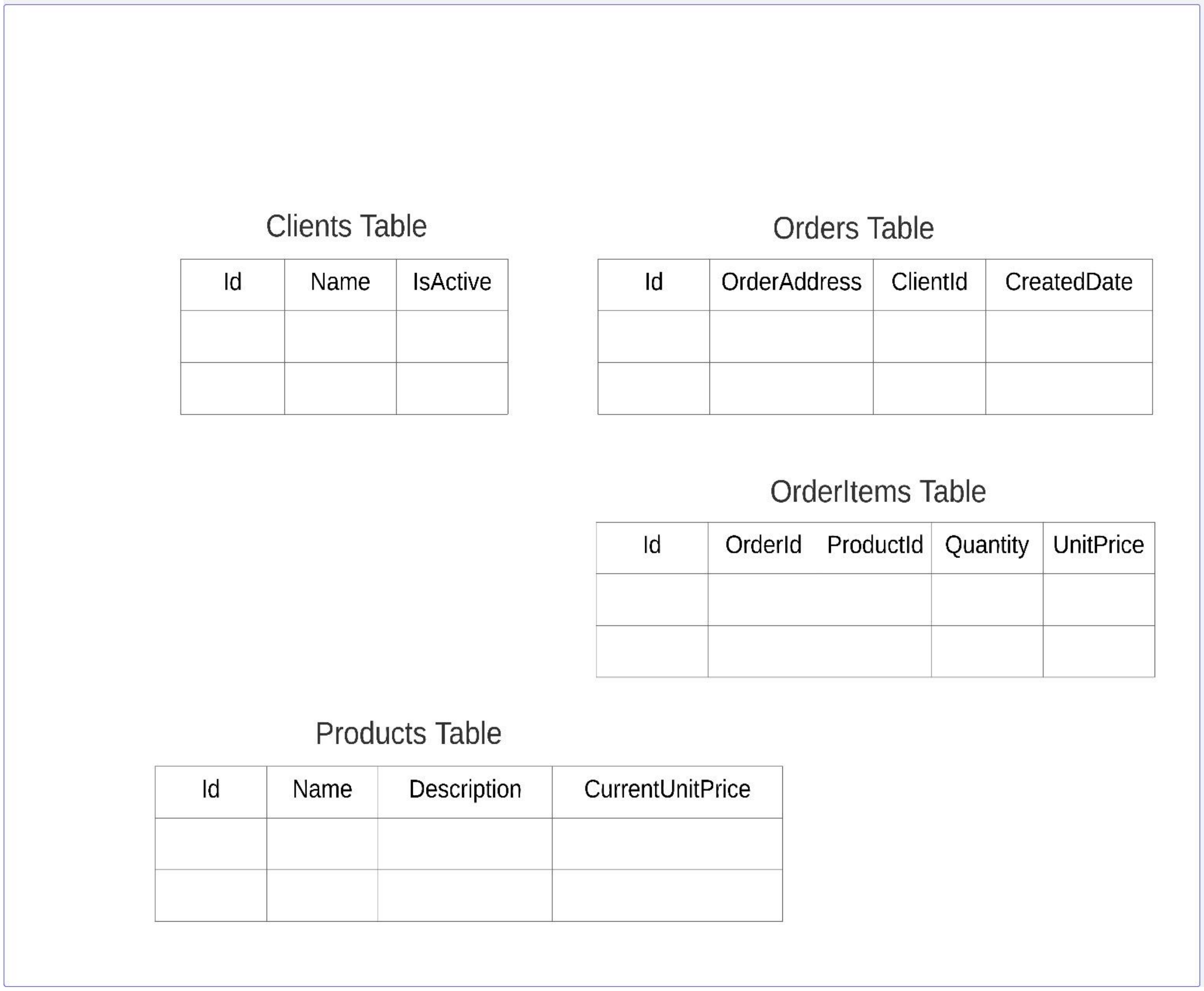
1. Different businesses need different databases , banking system needs database that different than social media app needs , you must choose the database that best suitable for your business domain.
2. Non-functional requirements is a very another important factor to choose your database , system of 1000 users needs database that different than system with 10,000,000 users.

SQL Databases

SQL databases also known as Relational Databases.

What is SQL Database ?

Database that store data on a tabular format (rows and columns)



Clients Table

Id	Name	IsActive

Orders Table

Id	OrderAddress	ClientId	CreatedDate

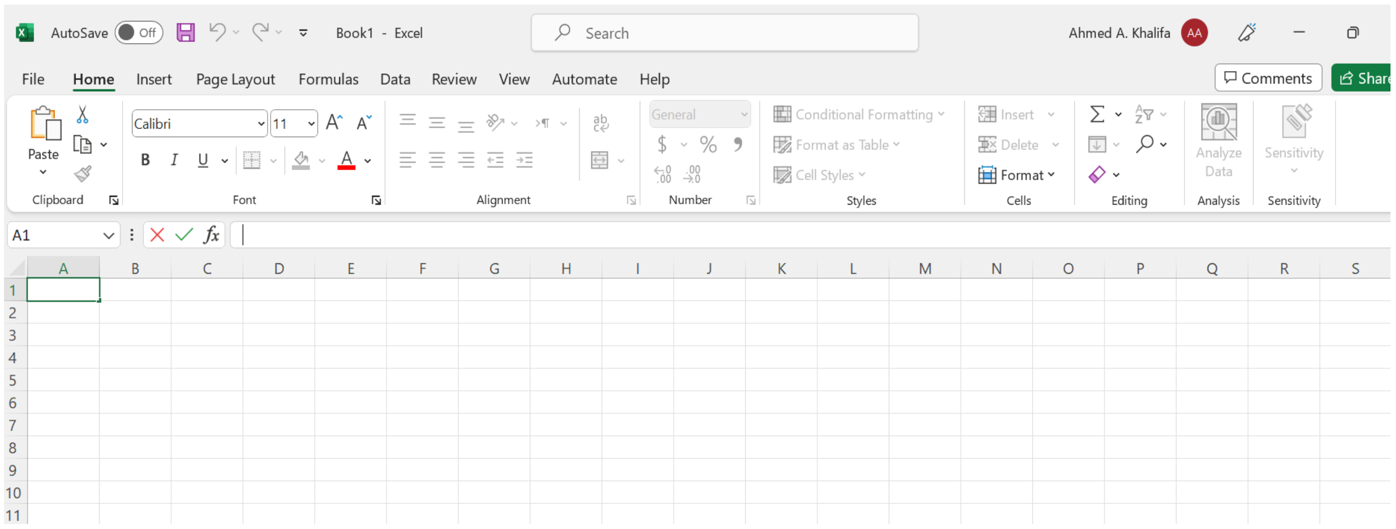
OrderItems Table

Id	OrderId ProductId	Quantity	UnitPrice

Products Table

Id	Name	Description	CurrentUnitPrice

Is Excel sheets can be SQL Database because it store data on tables (rows – columns) ?



Excel sheets is just a program to store data , **Not** any program or data store use tables, rows and columns will be SQL Database.

SQL Database has a set of standard conditions or properties that must be implemented and here list of the most important (not all) of them :

1- Relations.

2- Normalization.

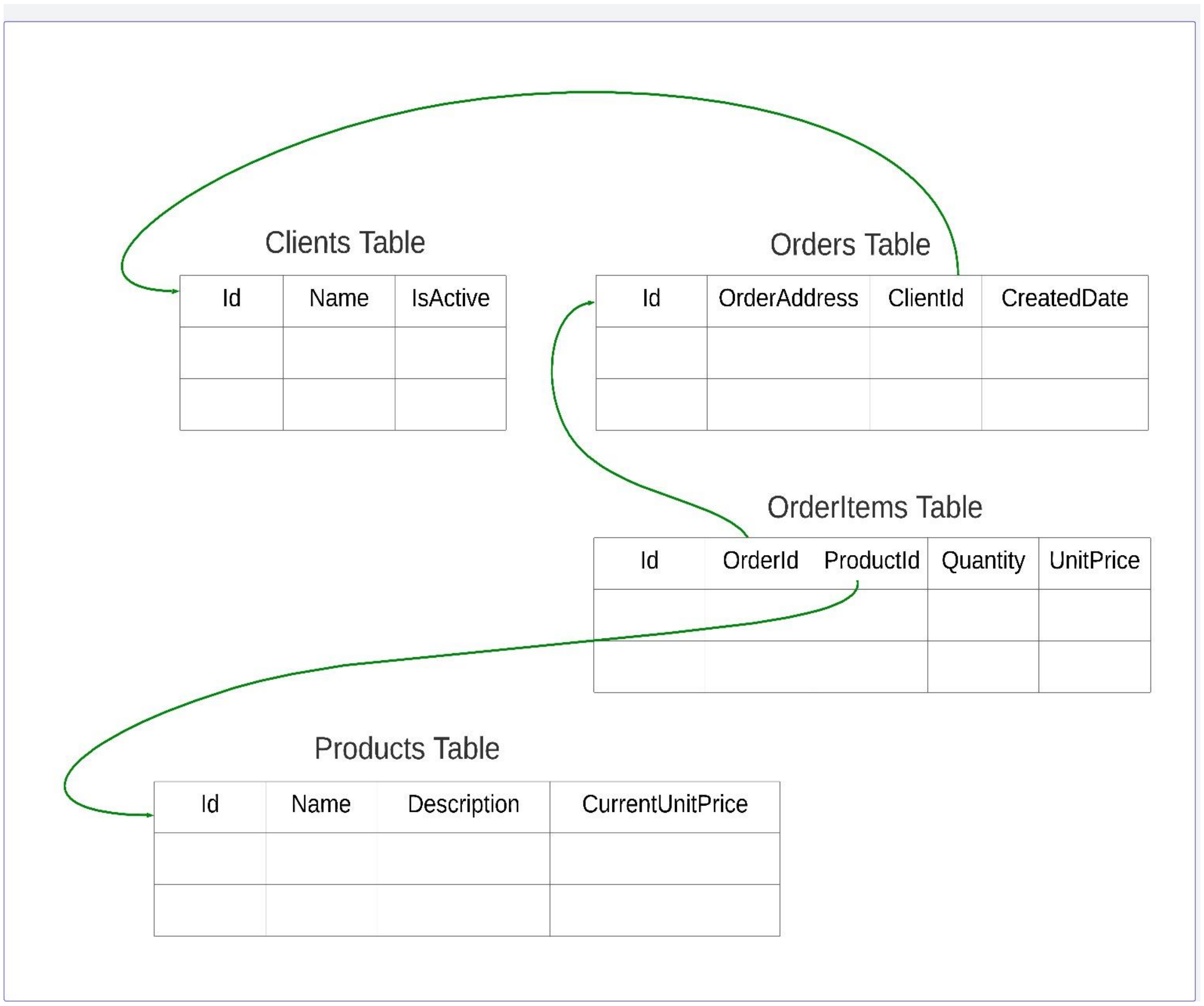
3- ACID.

We will discuss them in more details in next section.

SQL Database Properties :

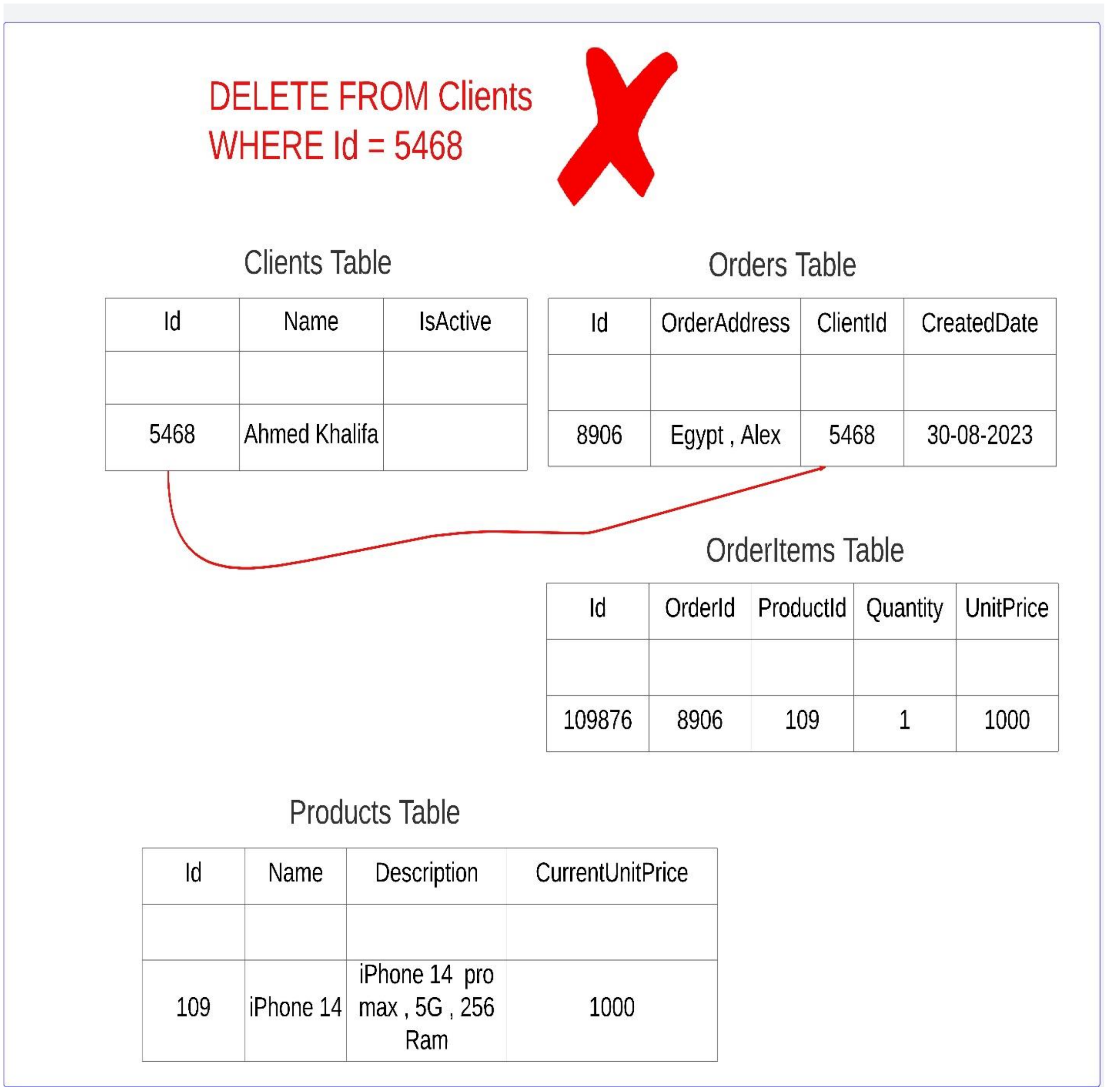
1- Relations :

The tables have relations between each other using foreign keys.



This relations help SQL database to achieve :

1- Data integrity for write operations (insert – update – delete) :
for example you can not delete client when he has orders and there reference to this client id in order table.



```
DELETE FROM Clients
WHERE Id = 5468
```

Clients Table

Id	Name	IsActive
5468	Ahmed Khalifa	

Orders Table

Id	OrderAddress	ClientId	CreatedDate
8906	Egypt , Alex	5468	30-08-2023

OrderItems Table

Id	OrderId	ProductId	Quantity	UnitPrice
109876	8906	109	1	1000

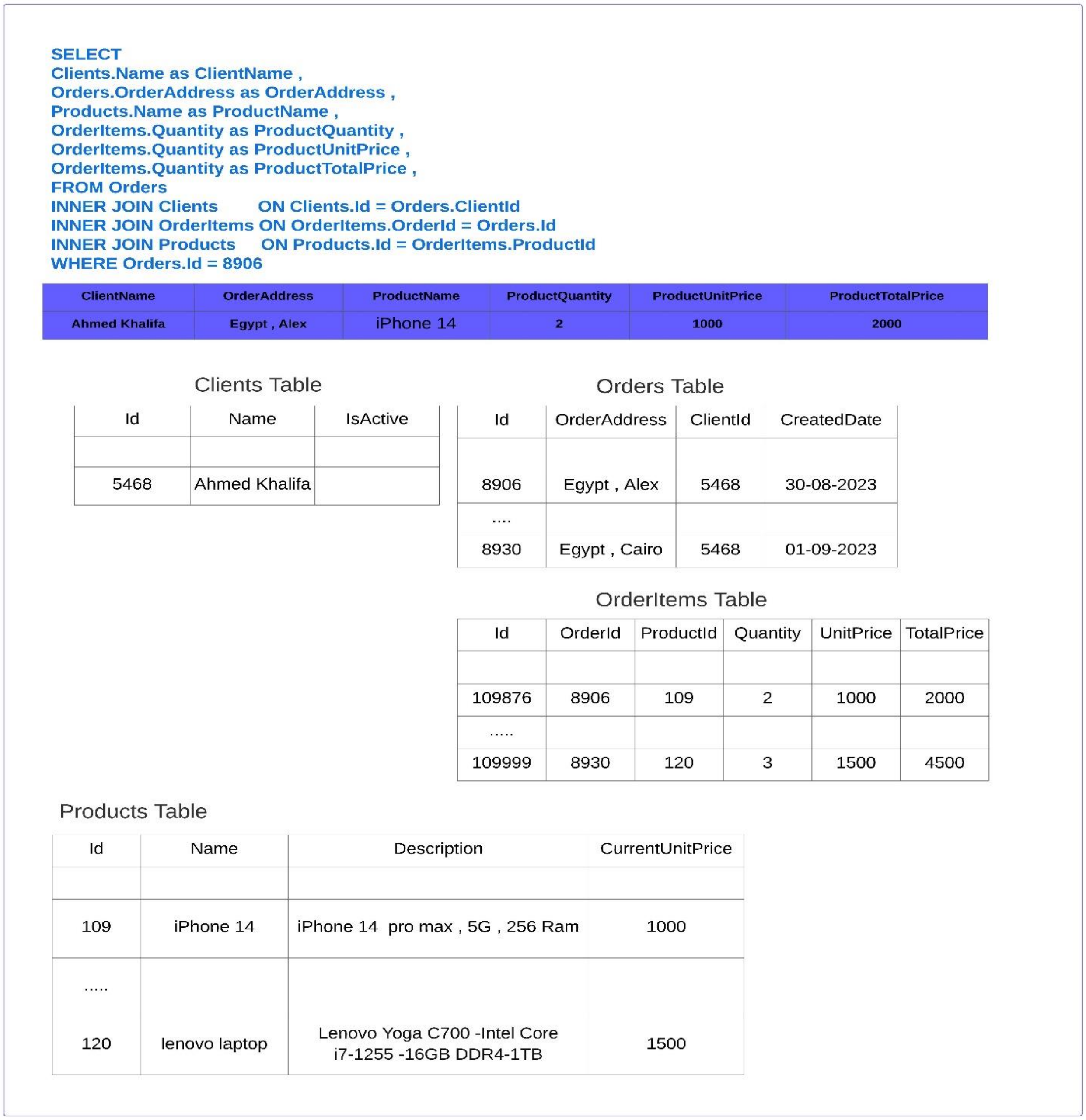
Products Table

Id	Name	Description	CurrentUnitPrice
109	iPhone 14	iPhone 14 pro max , 5G , 256 Ram	1000

2- Join query for read operations :

Join query is one of the main properties on SQL databases.

using Join query : SQL Database can read and display data from different tables in single query.



```
SELECT
Clients.Name as ClientName ,
Orders.OrderAddress as OrderAddress ,
Products.Name as ProductName ,
OrderItems.Quantity as ProductQuantity ,
OrderItems.Quantity as ProductUnitPrice ,
OrderItems.Quantity as ProductTotalPrice ,
FROM Orders
INNER JOIN Clients       ON Clients.Id = Orders.ClientId
INNER JOIN OrderItems ON OrderItems.OrderId = Orders.Id
INNER JOIN Products     ON Products.Id = OrderItems.ProductId
WHERE Orders.Id = 8906
```

ClientName	OrderAddress	ProductName	ProductQuantity	ProductUnitPrice	ProductTotalPrice
Ahmed Khalifa	Egypt , Alex	iPhone 14	2	1000	2000

Clients Table

Id	Name	IsActive
5468	Ahmed Khalifa	

Orders Table

Id	OrderAddress	ClientId	CreatedDate
8906	Egypt , Alex	5468	30-08-2023
....			
8930	Egypt , Cairo	5468	01-09-2023

OrderItems Table

Id	OrderId	ProductId	Quantity	UnitPrice	TotalPrice
109876	8906	109	2	1000	2000
.....					
109999	8930	120	3	1500	4500

Products Table

Id	Name	Description	CurrentUnitPrice
109	iPhone 14	iPhone 14 pro max , 5G , 256 Ram	1000
.....			
120	lenovo laptop	Lenovo Yoga C700 -Intel Core i7-1255 -16GB DDR4-1TB	1500

2- Normalization :

Normalization in SQL Databases aims to prevent data duplications.

Normalization has a lot of forms , but we will discuss only the general concept.

In the previous section (Relations) we see our database consist of a set of different tables that have relations between each other , if we need data from more than one table , we join this tables.

The question now :
why we design our database as a small tables as possible in development time and then joins them again in the runtime ?

It is better to design the database as single big table from scratch ?

The answer is summarized on two points :

1- For read operations :

Different queries need different data.

Example Query 1 : Get Order Details Of Id 8906

```
SELECT
Clients.Name as ClientName ,
Orders.OrderAddress as OrderAddress ,
Products.Name as ProductName ,
OrderItems.Quantity as ProductQuantity ,
OrderItems.Quantity as ProductUnitPrice ,
OrderItems.Quantity as ProductTotalPrice ,
FROM Orders
INNER JOIN Clients      ON Clients.Id = Orders.ClientId
INNER JOIN OrderItems ON OrderItems.OrderId = Orders.Id
INNER JOIN Products     ON Products.Id = OrderItems.ProductId
WHERE Orders.Id = 8906
```

ClientName	OrderAddress	ProductName	ProductQuantity	ProductUnitPrice	ProductTotalPrice
Ahmed Khalifa	Egypt , Alex	iPhone 14	2	1000	2000

Example Query 2 : Get All Orders Of Client 5468

```
SELECT
Clients.Name as ClientName ,
Orders.Id as OrderId ,
Orders.CreatedDate as OrderCreatedDate ,
FROM Orders
INNER JOIN Clients   ON Clients.Id = Orders.ClientId
WHERE Clients.Id = 5468
```

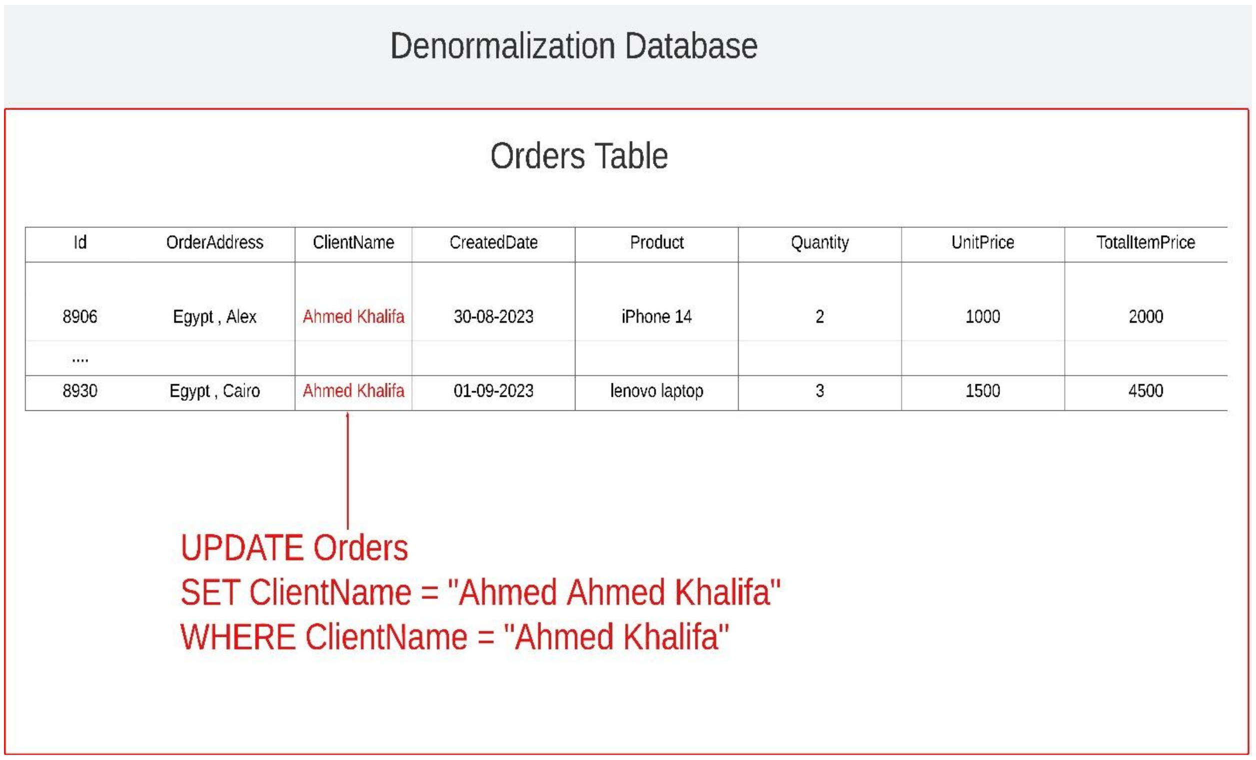
ClientName	OrderId	OrderCreatedDate
Ahmed Khalifa	8906	30-08-2023
Ahmed Khalifa	8930	01-09-2023

2- For write operations :

Prevent data duplication using normalization that can cause a lot of conflicts , performance issues , extra storge space for unnecessary data.

For example :
imagine our e-commerce database consist of only single table called orders table contain every thing.

Our client want to update or change his name , in denormalization database , the update command need to loop on all rows to update our client name.



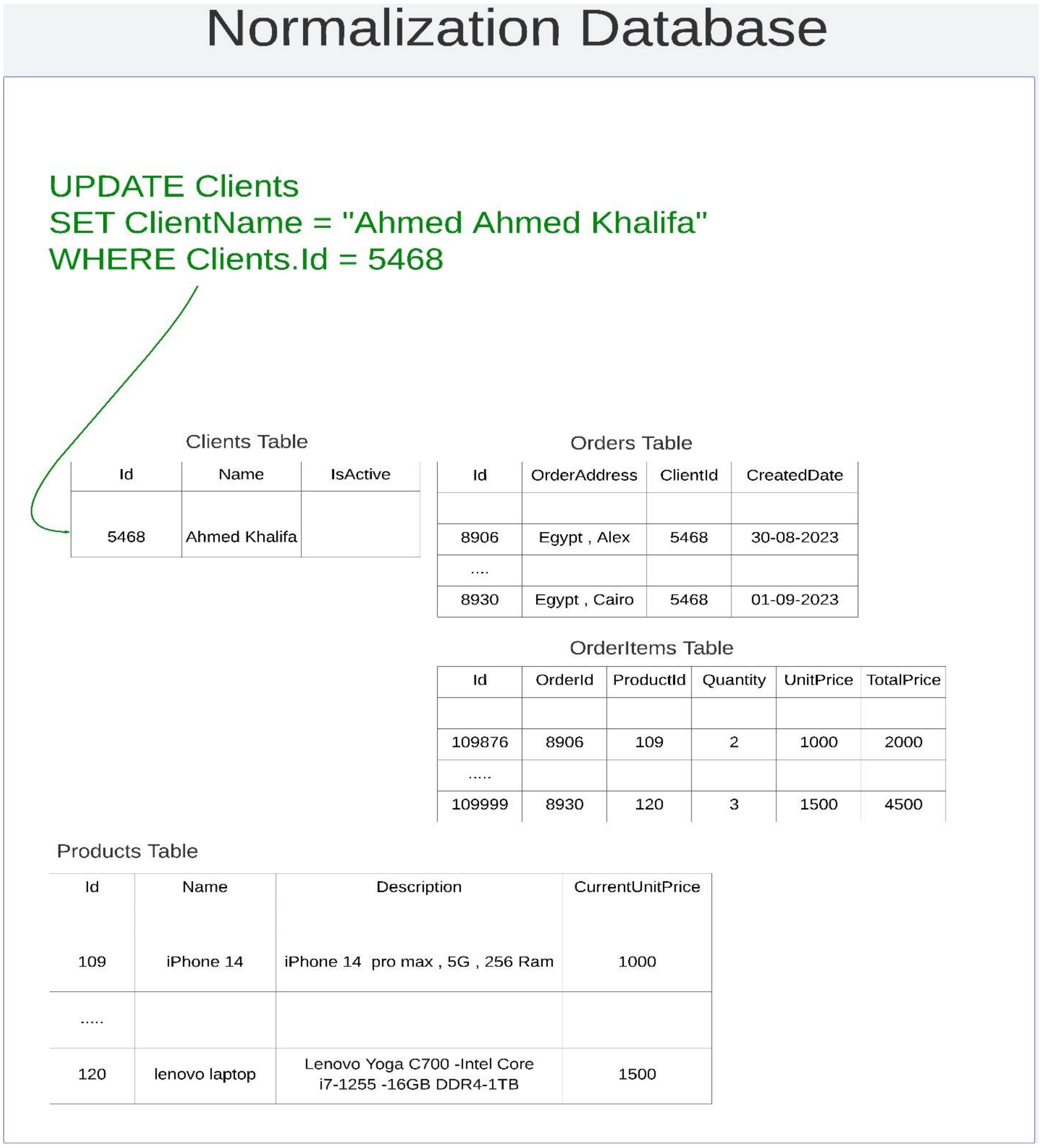
Denormalization Database

Orders Table

Id	OrderAddress	ClientName	CreatedDate	Product	Quantity	UnitPrice	TotalItemPrice
8906	Egypt , Alex	Ahmed Khalifa	30-08-2023	iPhone 14	2	1000	2000
....							
8930	Egypt , Cairo	Ahmed Khalifa	01-09-2023	lenovo laptop	3	1500	4500

```
UPDATE Orders
SET ClientName = "Ahmed Ahmed Khalifa"
WHERE ClientName = "Ahmed Khalifa"
```

But when database apply normalization , we need only to update single row , and any join query with this table or row will reflect the new updated value immediately.



Normalization Database

```sql
UPDATE Clients
SET ClientName = "Ahmed Ahmed Khalifa"
WHERE Clients.Id = 5468
```

Clients Table

Id	Name	IsActive
5468	Ahmed Khalifa	

Orders Table

Id	OrderAddress	ClientId	CreatedDate
8906	Egypt , Alex	5468	30-08-2023
....			
8930	Egypt , Cairo	5468	01-09-2023

OrderItems Table

Id	OrderId	ProductId	Quantity	UnitPrice	TotalPrice
109876	8906	109	2	1000	2000
.....					
109999	8930	120	3	1500	4500

Products Table

Id	Name	Description	CurrentUnitPrice
109	iPhone 14	iPhone 14 pro max , 5G , 256 Ram	1000
.....			
120	lenovo laptop	Lenovo Yoga C700 -Intel Core i7-1255 -16GB DDR4-1TB	1500

Is normalization responsibility for database or developers ?

Most of SQL Databases properties like ACID , Join query , data integrity are responsibility of database or RDBMS.

But normalization is responsibility o developers and their design.

Is Denormalization always bad ?

In some cases denormalization can be good but developers must take care of the effect sides of it.

Denormalization can good for :
1- some business cases for historical data :
if you note in our database design Products table has column called CurrentUnitPrice and OrderItems Table has column called UnitPrice , this because product price can change on the future , the old orders with this product must keep the old price.

2- solve performance issues when your query has a depth or a lot of joins , you can duplicate the column on one of the highest level table in join

query and prevent join with that low level table ,
this solution can be good when your duplicated column is rarely updated.

in general this solution must be used only in a very necessary cases and must not be common in your database.