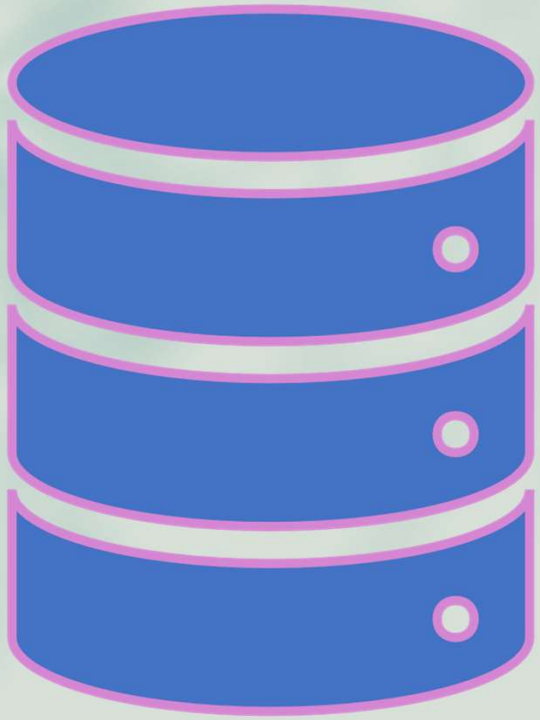
DATABASE TECHNOLOGIES

Tânia de Lima Santos

Outubro 2023

Index



1. Database Fundamentals

Database Fundamentals - Topics

- Database concept
- Relation and non-relational
- Major providers

Database Fundamentals – Database concept

- A database is:
  - An organized collection of structured information
  - Typically stored electronically in a computer system
  - The data is typically modeled in rows and columns in a series of tables to make processing and data querying eficiente
  - Allows for data to be easily accessed, managed, modified updated, controlled and organized
  - Most database use structured query language (SQL) for writing and querying data

Database Fundamentals – Database concept

- What is a Database Management System?
  - A database management system (DBMS) is a collection of programs that enables users to create and maintains a database

- Advantages of a DBMS:
  - Allow concurrency
  - Control security
  - Maintain data integrity
  - Provide for backup and recovery
  - Control redundancy
  - Allow data independency
  - Provide non-procedural query language
  - Perform automatic query optimization

Database Fundamentals – Relational vs Non-Relational

- Relational databases:
  - Items in a relational database are organized as a set of tables with columns and rows
  - Relational database technology provides the most eficiente and flexible way to access structured information

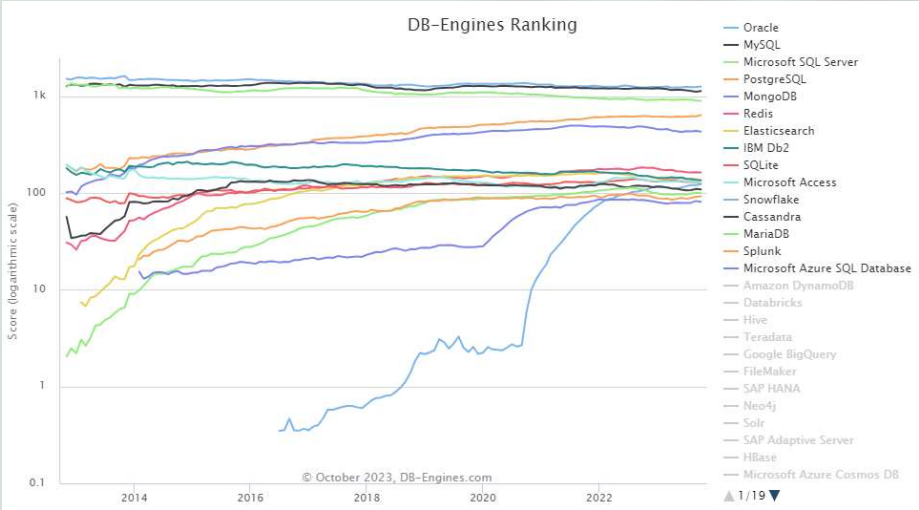
- NoSQL databases:
  - NoSQL, or nonreational database, allows unstructured and semistructured data to be stored and manipulated (in contrast to a relational database, which defines how all data inserted into the database must be composed)
  - NoSQL databases grew popular as web applications became more common and more complex

Database Fundamentals – Relational vs Non-Relational

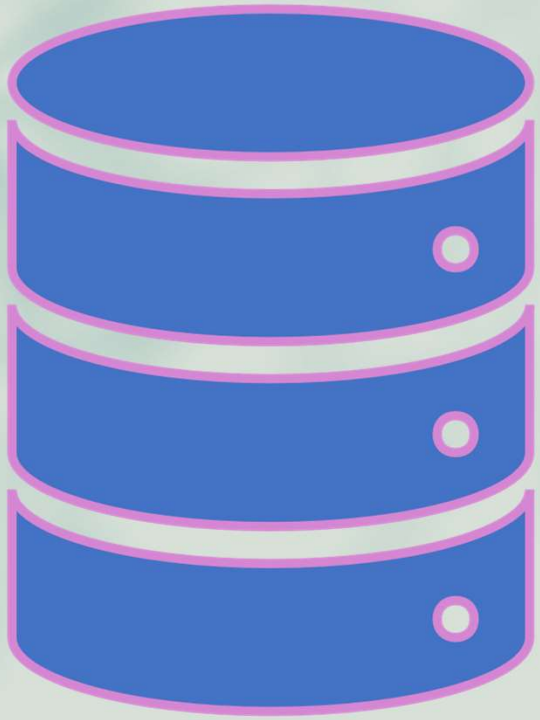
- SQL vr NoSQL:
  - SQL databases are tables based whereas NoSQL databases can be document based, key-value pairs, graph databases.
  - SQL databases are vertically scalable (ram,…) while NoSQL databases are horizontally (servers) scalable.
  - SQL databases have a predefined schema whereas NoSQL databases use dynamic schema for unstructured data.
  - SQL requires specialized DB hardware for better performance while NoSQL uses commodity hardware.
  - SQL database examples: MySql, Oracle, Sqlite, Postgres and MS-SQL.
  - NoSQL database examples: MongoDB, BigTable, Redis, RavenDb, Cassandra, Hbase, Neo4j and CouchDb

Database Fundamentals – Major providers

Rank			DBMS	Database Model	Score		
Oct 2023	Sep 2023	Oct 2022			Oct 2023	Sep 2023	Oct 2022
1.	1.	1.	Oracle	Relational, Multi-model	1261.42	+20.54	+25.05
2.	2.	2.	MySQL	Relational, Multi-model	1133.32	+21.83	-72.06
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	896.88	-5.34	-27.80
4.	4.	4.	PostgreSQL	Relational, Multi-model	638.82	+18.06	+16.10
5.	5.	5.	MongoDB	Document, Multi-model	431.42	-8.00	-54.81
6.	6.	6.	Redis	Key-value, Multi-model	162.96	-0.72	-20.41
7.	7.	7.	Elasticsearch	Search engine, Multi-model	137.15	-1.84	-13.92
8.	8.	8.	IBM Db2	Relational, Multi-model	134.87	-1.85	-14.79
9.	9.	↑ 10.	SQLite	Relational	125.14	-4.06	-12.66
10.	10.	↓ 9.	Microsoft Access	Relational	124.31	-4.25	-13.85



Source: DB-Engines Ranking - popularity ranking of database management systems (db-engines.com)



2.
Data Models

Data Models
- Topics

- Table, View e MView
- PK, FK, Index
- Schema
- Functions
- Model definition
- Types of Data Models

Data Models – Table, View and MView

- Table
  - Dataset with data from the same subject
  - Made up of rows and columns
  - Each row is uniquely identified by a primary key (in a relational DB)
  - Each column stores a specific type of data

ID	Title	ISBN	Author	Publishing...
1	1984	2343454895456	George Orwell	04/12/1979
2	Anna Karenina	1234548485843	Leo Tolstoy	07/11/1998
4	The Adventures of I	3450345345443	Mark Twain	08/11/1999
5	Ulysses	9944933003232	James Joyce	06/05/2010
8	War and Peace	0944344903312	Leo Tolstoy	08/11/2001
11	The Brothers Karan	9003940397271	Doso	04/07/2012
12	On the Road	0459450444310	Jack Kerouac	30/12/2005
15	The Metamorphosi	2003948930545	Franz Kafka	09/03/1976
16	The Illiad	9449039333923	Homer	05/07/1998
17	The Odyssey	8409404850139	Homer	06/08/1999

Data Models– Table, View and MView

- View
  - Looks like a table, but it's a virtual table (only exists while used)
  - Made up of rows and columns
  - Changes applied to the data in na underlying table are reflected in the view
  - Usually a subset of a table or result from joining existing tables

ID	Title	ISBN	Author	Publishing...
1	1984	2343454895456	George Orwell	04/12/1979
2	Anna Karenina	1234548485843	Leo Tolstoy	07/11/1998
4	The Adventures of I	3450345345443	Mark Twain	08/11/1999
5	Ulysses	9944933003232	James Joyce	06/05/2010
8	War and Peace	0944344903312	Leo Tolstoy	08/11/2001
11	The Brothers Karan	9003940397271	Doso	04/07/2012
12	On the Road	0459450444310	Jack Kerouac	30/12/2005
15	The Metamorphosi	2003948930545	Franz Kafka	09/03/1976
16	The Illiad	9449039333923	Homer	05/07/1998
17	The Odyssey	8409404850139	Homer	06/08/199[illegible]

Data Models– Table, View and MView

- Materialized View
  - Unlike views, mviews are stored in the table or disk (so have better performance)
  - Can be indexed as a table
  - Made up of rows and columns
  - Can define the refresh hour and frequency
  - Changes applied to the data in an underlying table are reflected in the view after a refresh
  - Usually a subset of a table or result from joining existing tables

ID	Title	ISBN
1	1984	2343454895456
2	Anna Karenina	1234548485843
4	The Adventures of I	3450345345443
5	Ulysses	9944933003232
8	War and Peace	0944344903312
11	The Brothers Karan	9003940397271
12	On the Road	0459450444310
15	The Metamorphosi	2003948930545
16	The Illiad	9449039333923
17	The Odyssey	8409404850139

Data Models – PK, FK & Index

- Primary Key (PK)
  - One column (or combination of columns) that uniquely identify each row
  - Contains a unique value that uniquely identify each row
  - Cannot be null
  - Can also be a column that is specifically generated by the database according to a defined sequence

Data Models – PK, FK & Index

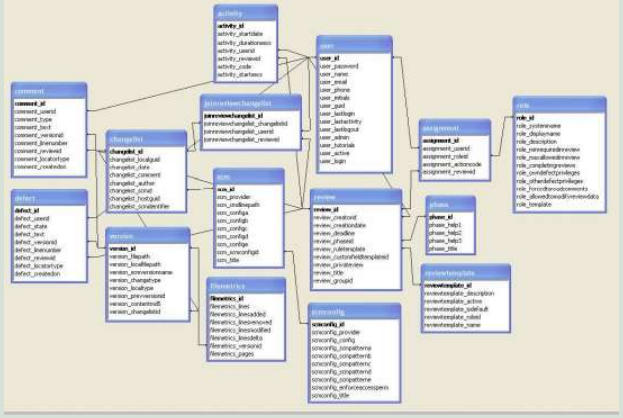
- Foreign Key (FK)
  - aka "referencing key"
  - a key used to link two tables
  - a column or a combination of columns, matching a PK on other table.
  - The relationship between 2 tables matches the PK in one with a FK in the other

Data Models – PK, FK & Index

- Index
  - Indexes are used to find rows quickly
  - Without an index, search begin by the first row and then read through the entire table
  - Most common: PRIMARY KEY, UNIQUE, INDEX

Data Models – Schema

- Schema
  - Represents how the data is organized in tables and how tables link between them
  - Abstraction to logically group objects such as tables, joins and keys
  - A schema is the skeleton of database
  - Is the design for the data modified to fit in a specific database technology
  - Is about the implementation of the database



Data Models – Functions

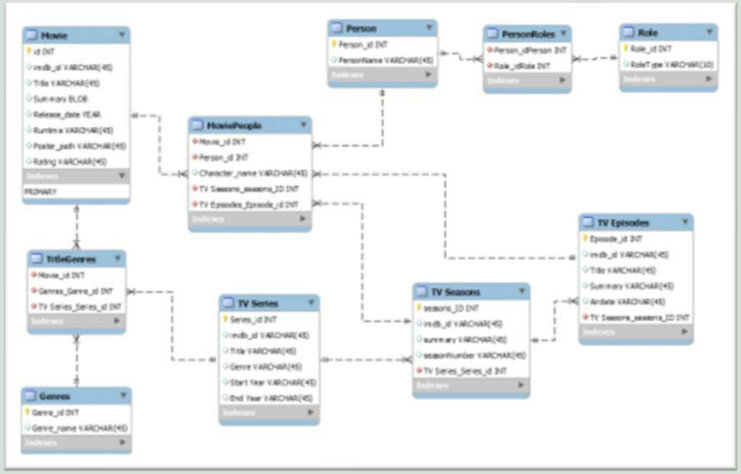
- Function
  - Allows to encapsulate operations that can take several steps in a single functions within the database
  - Allows reusing code blocks
  - Can be created in several languages (eg. SQL, C, Python)
  - https://www.postgresql.org/docs/9.1/sql-createfunction.html

```
CREATE [OR REPLACE] FUNCTION function_name (arguments)
RETURNS return_datatype AS $variable_name$
  DECLARE
    declaration;
    [...]
  BEGIN
    < function_body >
    [...]
    RETURN { variable_name | value }
  END; LANGUAGE plpgsql;
```

```
CREATE OR REPLACE FUNCTION increment(i integer)
RETURNS integer AS $$
BEGIN
  RETURN i + 1;
END;
$$ LANGUAGE plpgsql;
```

Data Models – Model definition

- Database model
  - The definition of the database
  - The design for the data (no matter the storage technology)
  - Gives a conceptual understanding of how is structured in a database

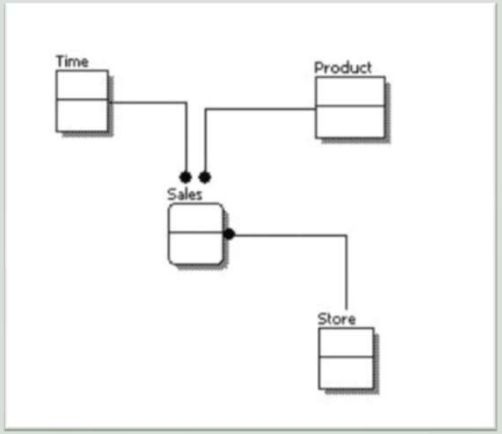


Data Models – Types of Data Model

- Data Model concepts:
  - Entity: A real-world thing (eg. Customer)
  - Attribute: Characteristics or properties of na entity (eg. Age)
  - Relationship: Dependency or association between two entities (eg. Related to Product table)

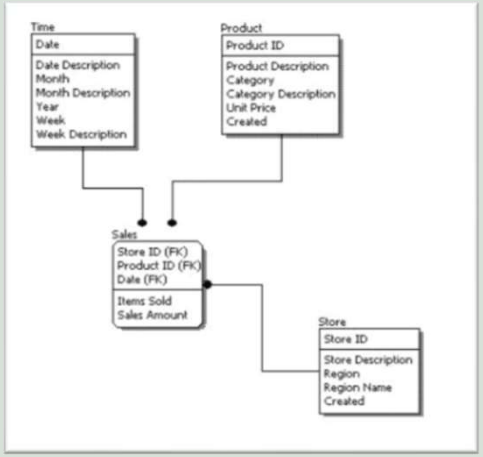
Data Models – Types of Data Model

- Conceptual:
  - High-level = least detail
  - Designed for a business audience (less technical)
  - Establish the entities, some attributes, and high level relationships
  - Developed independently of hardware specificatios



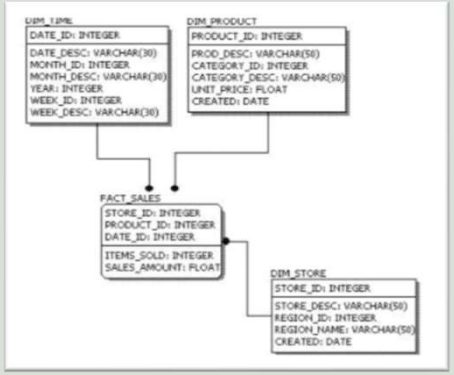
Data Models – Types of Data Model

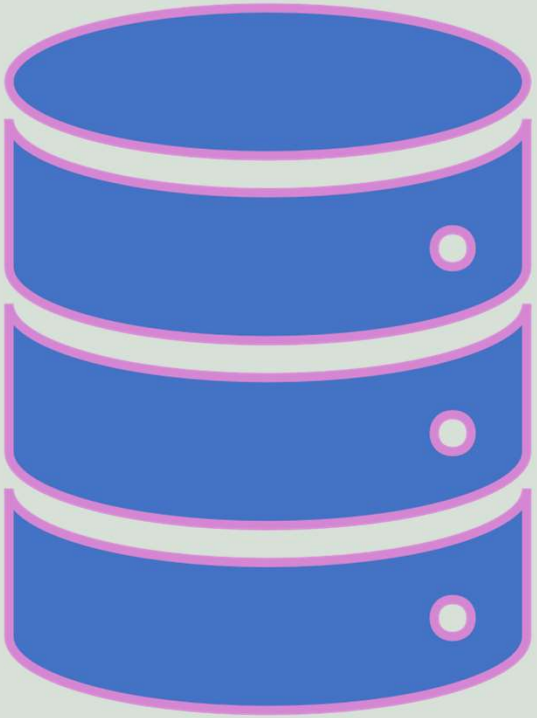
- Logical:
  - Add entity types, data attributes and relationships between entities
  - Include most business detail
  - Describes data requirements from the business point of view
  - Fully-attributed data model that is independent technology details
  - Data attributes will typically have datatypes



Data Models – Types of Data Model

- Physical:
  - The internal schema database design
  - Technically aligned to a platform and capable of generating a first cut database schema.
  - Fully-attributed data model that is dependent upon a specific version of a data persistence technology
  - Include physical objects such as views, primary key constraints, foreign key constraints, indexes





3. SQL

SQL - Topics

Key Concepts

PostgreSQLKey

Exercises

SQL – Key Concepts

- What can we do?
  - Data interrogation
  - Data definition
  - Data manipulation
  - Data control

SQL – Key Concepts

- Select statement… *the beginning of your sql journey*!

- SELECT statement has the following clauses
  - WHERE
  - GROUP BY
  - DISTINCT
  - ORDER BY
  - LIMIT
  - FETCH
  - HAVING
  - JOIN (INNER JOIN, LEFT JOIN, FULL OUTER JOIN, CROSS)
  - UNION
  - INTERSECT
  - EXCEPT

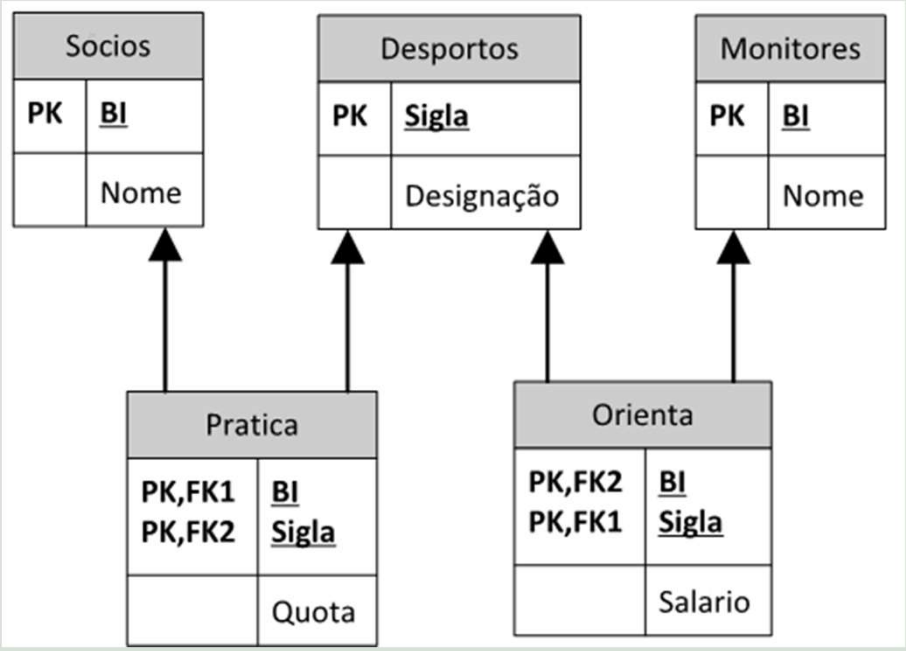
SQL– Key concepts

- Operators:

Operator	Action
=	Tests for equality
!=	Tests for inequality
<	Tests for less-than
>	Tests for greater-than
<=	tests for less-than or equal-to
>=	tests for greater-than or equal-to
BETWEEN	tests whether a value lies within a given range
IN	tests whether a row's value is contained in a set of specified values
EXISTS	tests whether rows exist, given the specified conditions
LIKE	tests whether a value matches a specified string
IS NULL	tests for NULL values
IS NOT NULL	tests for all values other than NULL

SQL – Example of a Relational Model

- Gym example:



SQL – Example of a Relational Model

- Gym example:

Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Monitores

BI	Nome
9876	Luís
1234	Joana

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

SQL - Select

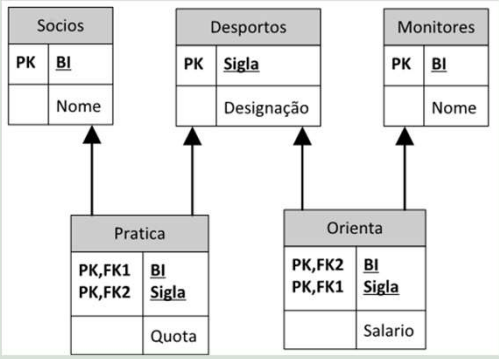
- **Used to show the information needed**

```
Select col1, col2,…
From table1
```

SQL - Select

- **Used to show the information needed**

1. Show all the information about members



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

Monitores

BI	Nome
9876	Luís
1234	Joana

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

SQL - Select

- **Used to show the information needed**

1. Show all the information about all partners

select * ***from*** *socios;*

Result

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

SQL - Distinct

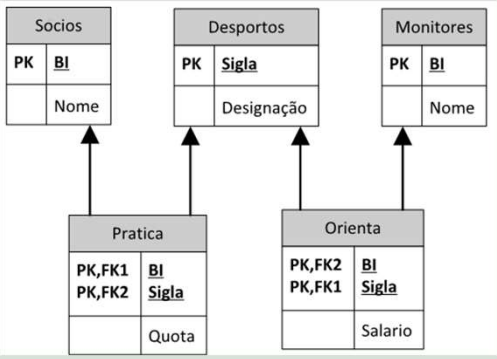
- **Remove Duplicates**

```
Select distinct col1, col2
From table1
```

SQL - Distinct

- **Remove Duplicates**

2. Select the acronyms of the sports that have participants



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Monitores

BI	Nome
9876	Luís
1234	Joana

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

SQL - Distinct

- **Remove Duplicates**

2. Select the acronyms of the sports that have participants

select *sigla*
from *pratica;*

Results

Sigla
AE
KB
KB
NT
KB
NT
AE
KB

select distinct *sigla*
from *pratica;*

Results

Sigla
AE
KB
NT

SQL – Where

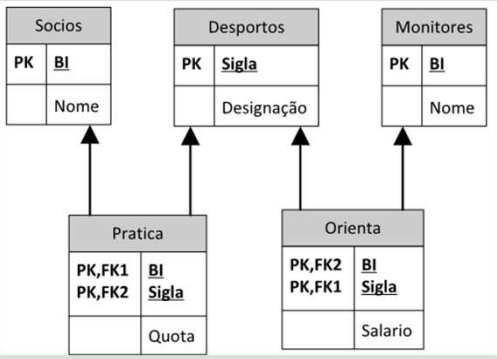
- **Used to filter rows**

```
Select col1, col2
From table1
Where <condition>
```

SQL – Where

- **Used to filter rows**

3. List member's BI number and name, whose BI number is less than 6000



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Monitores

BI	Nome
9876	Luís
1234	Joana

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

SQL – Where

- **Used to filter rows**

3. List member's BI number and name, whose BI number is less than 6000

select *bi, nome*

from *socios*

where *bi < 6000;*

Results

BI	Nome
5819	Rui
4526	Nuno
3955	Rita

SQL - Like

- **Search for strings**

Compare strings with **like** :

% look for any string with 0 or more characters :

name **like** ‘M%' (Oracle, MySQL)

'Marina’ will be na output

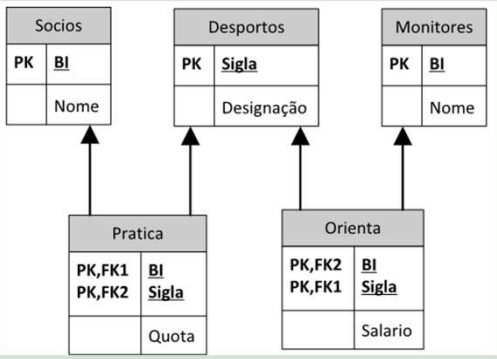
Compare strings with **=**, looks for the exact match :

name **=** 'M%’ is true if the name is 'M%'

SQL – Like

- **Search for strings**

4. List the member's BI number and name which member's name start with 'R' or BI number is between 4000 and 5000.



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Monitores

BI	Nome
9876	Luís
1234	Joana

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

SQL – Like

- **Search for strings**

4. List the member's BI number and name which member's name start with 'R' or BI number is between 4000 and 5000.

select *bi, nome*

from *socios*

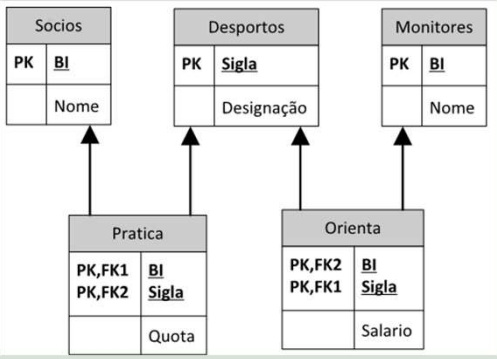
where *nome* ***like*** *'R%'* ***or*** *bi* ***between*** *4000* ***and*** *5000;*

Results

BI	Nome
5819	Rui
4526	Nuno
3955	Rita

SQL – Arithmetic Espressions

5. What would be the salaries above 40€ if we increase 20%?



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Monitores

BI	Nome
9876	Luís
1234	Joana

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

SQL – Arithmetic Espressions

5. What would be the salaries above 40€ if we increase 20%?

select *bi, sigla, salario*1.2* ***as*** *"Novo salário"*

from *orienta*

where *salario*1.2 > 40*

Results

BI	Sigla	Novo Salario
1234	KB	48
9876	AE	42

SQL - In

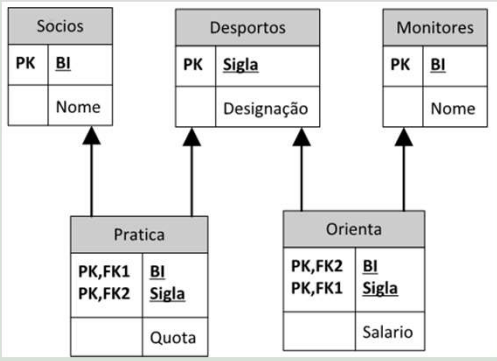
- **Value belong to a set**

6. List the member's ID and sport's acronyms, which exercise aerobics or swimming

SQL - In

- **Value belong to a set**

6. List the member's ID and sports acronyms which exercise aerobics or swimmers



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

Monitores

BI	Nome
9876	Luís
1234	Joana

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

SQL - In

- **Value belong to a set**

6. List the member's ID and sports acronyms which exercise aerobics or swimmers

```
select bi, sigla
from pratica
where sigla in ('AE', 'NT');

select bi, sigla
from pratica
where sigla='AE' or sigla='NT'
```

Results

BI	Sigla
3955	AE
3955	NT
4526	NT
6078	AE

SQL– Order by

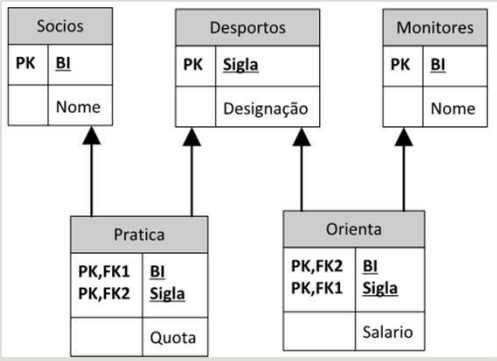
- **Output sorting**

```
Select col1, col2
From table1
Where <condition>
Order by col_1;
```

SQL– Order by

- **Output sorting**

7. Get a list of sports acronyms and member's BI sorted ascending by acronym and descending by BI



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

Monitores

BI	Nome
9876	Luís
1234	Joana

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

SQL – Order by

- **Output sorting**

7. Get a list of sports acronyms and member's BI sorted ascending by acronym and descending by BI

select *sigla, bi*

from *pratica*

order by *sigla , bi* ***desc***

Results

Sigla	BI
AE	6078
AE	3955
KB	9876
KB	5819
KB	4526
KB	3955
NT	4526
NT	3955

SQL – Aggregation Functions

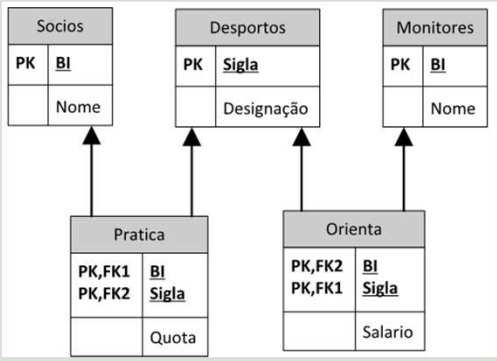
- **Aggregate rows and calculate values**

```
Select col1, col2,…, count(colx)
From table1
Where <condition>
Group by col1, col2, …;
```

SQL – Aggregation Functions

- **Aggregate rows and calculate values**

8. Get the average salary, number of salaries and total salaries, as well as the highest and lowest salary



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

Monitores

BI	Nome
9876	Luís
1234	Joana

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

SQL – Aggregation Functions

- **Aggregate rows and calculate values**

8. Get the average salary, number of salaries and total salaries, as well as the highest and lowest salary

```
select
    avg(salario) as mean,
    count(*) as number,
    sum(salario) as total,
    max(salario) as max,
    min(salario) as min
from orienta
```

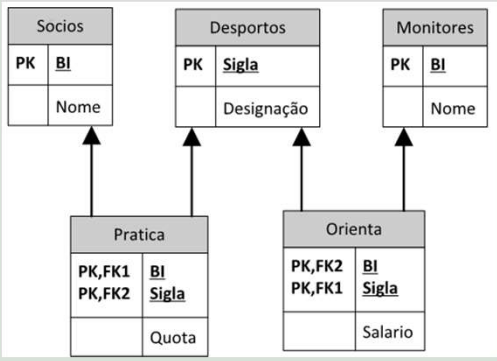
Results

Mean	Number	Total	Max	Min
33,75	4	135	40	30

SQL – Group By

- **Aggregate rows and calculate values**

9. Calculate the total amount paid by each member (bi and value)



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

Monitores

BI	Nome
9876	Luís
1234	Joana

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

SQL – Group By

- **Aggregate rows and calculate values**

9. Calculate the total amount paid by each member (bi and value)

select *bi*, ***sum****(quota)*

from *pratica*

group by *bi*

Results

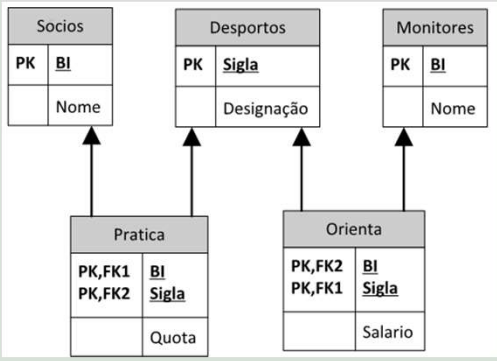
BI	Sum(quota)
3955	75
4526	50
5819	30
6078	25
9876	0

It is essential to have in the *group by* all the non-aggregated columns is the *select* statement

SQL – Group by with filters

- **Aggregate rows and calculate values**

10. Calculate the total amount paid by each member, but only for cases above 40€



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Monitores

BI	Nome
9876	Luís
1234	Joana

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

SQL – Group by with filters

- **Aggregate rows and calculate values**

10. Calculate the total amount paid by each member, but only for cases above €40

```
select bi, sum(quota)
from pratica
where sum(quota) > 40
group by bi
```

SQL – Group by & Having

- **Aggregate rows and calculate values**

10. Calculate the total amount paid by each member, but only for cases above 40€

~~***select*** *bi*, ***sum***(*quota*)~~
~~***from*** *pratica*~~
~~***where sum***(*quota*) > *40*~~
~~***group by*** *bi*~~

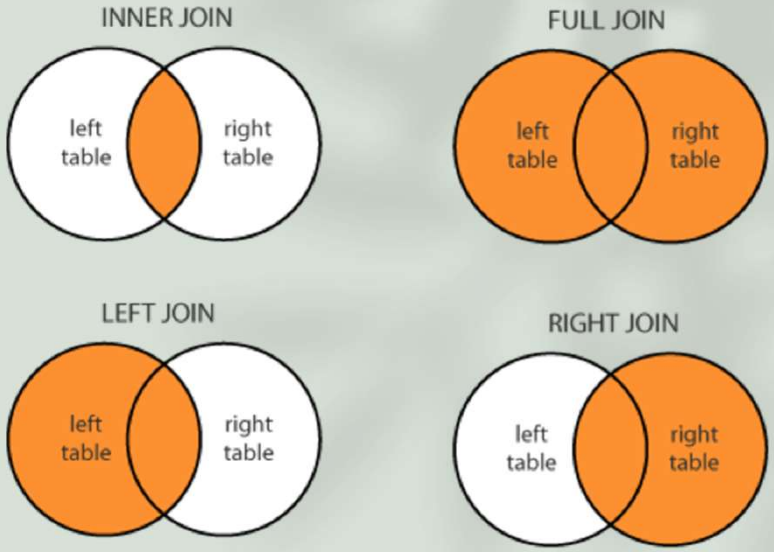
select *bi*, ***sum***(*quota*)
from *pratica*
group by *bi*
having sum(*quota*) > *40*

having selects rows from aggregation like ***where*** selects rows from base table

SQL – Joins

Most common types of joins:

- (Inner) join – only returns records that have matching values in both tables
- Left join – returns all records from the left table, and the matched records from the right table
- Right join – returns all records from the right table, and the matched records from the left table;

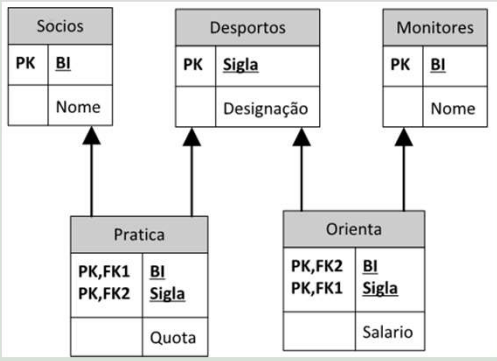



```
Select *
From table1 t1
(left/right) Join table2 t2 on t1.col1 = t2.col3
```

SQL – JOINS

Join tables

11. List the member's name and sport's acronym, who practice aerobics and swimming



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Monitores

BI	Nome
9876	Luís
1234	Joana

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

SQL – JOINS

- **Join tables**

11. List the member's name and sport's acronym, who practice aerobics and swimming

select *nome, sigla*

from *socios*

join *pratica* ***on*** *socios.bi=pratica.bi*

where *sigla* ***in*** *('AE', 'NT')*

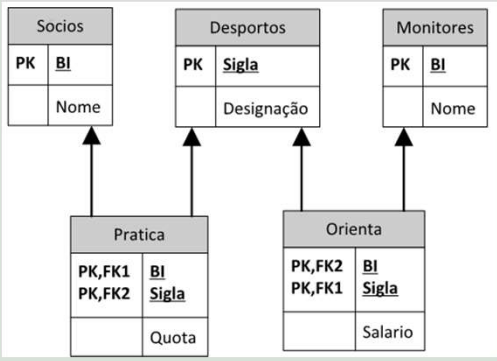
Results

Nome	Sigla
Ana	AE
Nuno	NT
Rita	NT
Rita	AE

SQL – JOINS

Join tables

12. List the member's name and sport's name, who practice aerobics and swimming



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Monitores

BI	Nome
9876	Luís
1234	Joana

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

SQL – JOINS

- **Join tables**

12. List the member's name and sport's name, who practice aerobics and swimming

select *s.nome, d.designação*
from *socios s*
join *pratica p* ***on*** *s.bi=p.bi*
join *desportos d* ***on*** *p.sigla = d.sigla*
where *p.sigla* ***in*** *('AE', 'NT')*

Results

Nome	Designação
Ana	Aeróbica
Nuno	Natação
Rita	Natação
Rita	Aeróbica

SQL – UNION & INTERSECT

- **Union / Intersect tables**

UNION Statement

```
Select col1, col2
From table1
Union (all)
Select col3, col4
From table2;
```

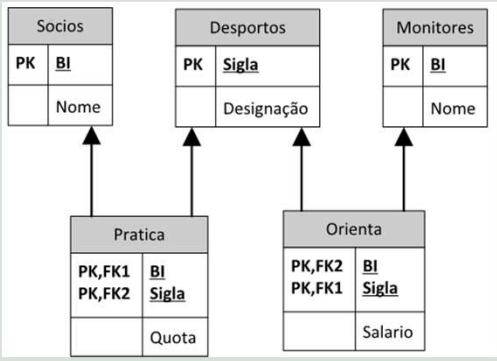
INTERSECT Statement

```
Select col1, col2
From table1
Intersect
Select col3, col4
From table2;
```

SQL – UNION

- **Union / Intersect tables**

13. List all names from members and monitors?



Socios

BI	Nome
6078	Ana
5819	Rui
4526	Nuno
3955	Rita
9999	José

Desportos

Sigla	Designação
KB	Kickbox
NT	Natação
AE	Aeróbica

Pratica

BI	Sigla	Quota
6078	AE	25
5819	KB	30
4526	KB	30
4526	NT	20
3955	KB	30
3955	NT	20
3955	AE	25
9876	KB	0

Monitores

BI	Nome
9876	Luís
1234	Joana

Orienta

BI	Sigla	Salario
1234	KB	40
1234	NT	30
9876	NT	30
9876	AE	35

SQL – UNION

13. List all names from members and monitors?

(select *nome*

from *socios)*

Union

(select nome

From *monitores)*

SQL– INTERSECT

14. List the common names from members and monitors?

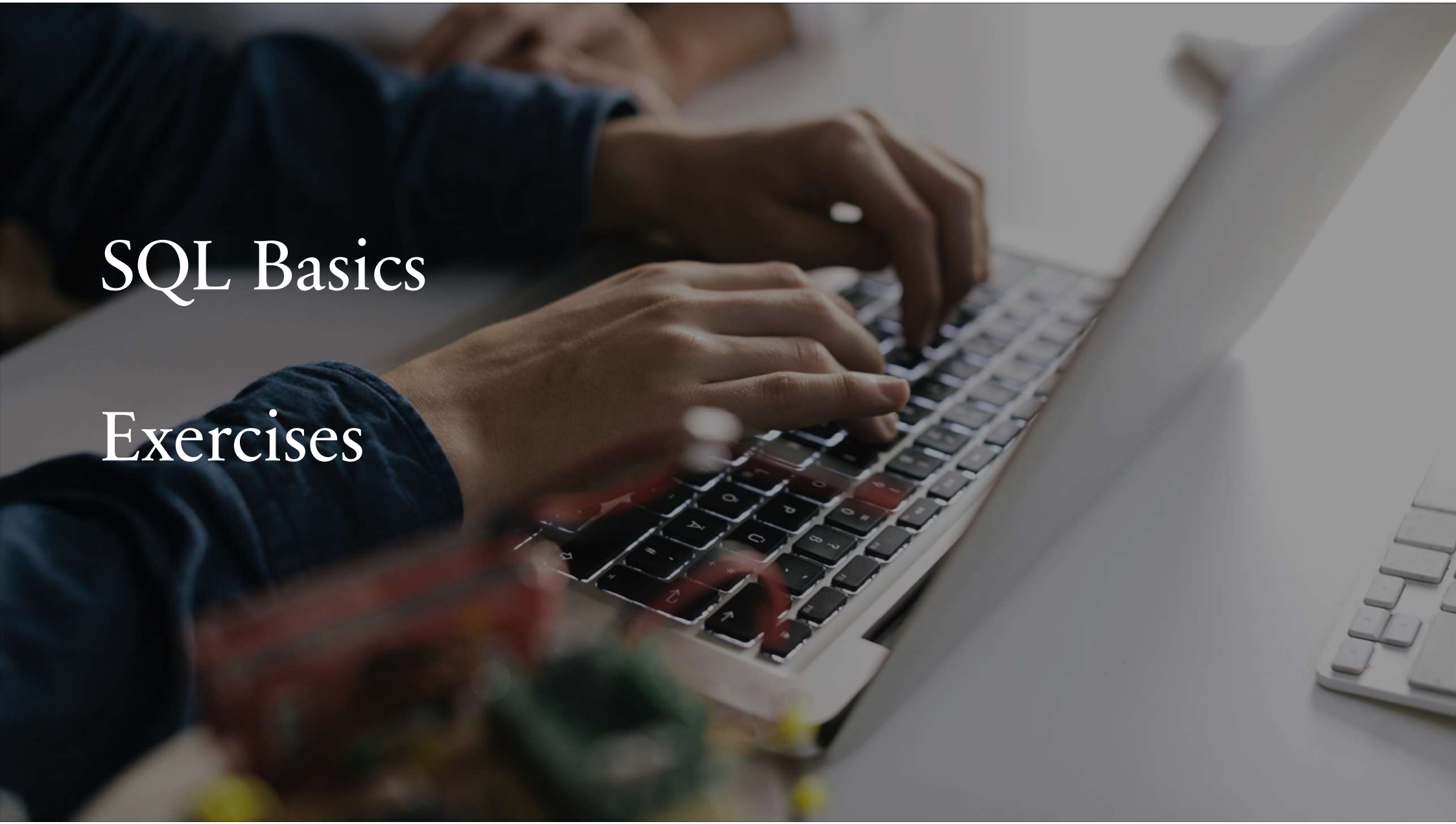
(select *nome*

from *socios)*

intersect

(select nome

From *monitores)*



SQL Basics

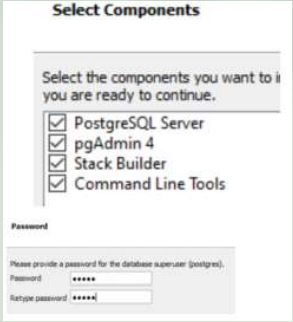
Exercises

SQL – PostgreSQL

- PostgreSQL:
  - Also known as postgres
  - Not only for academic purposes
  - Relational database
  - (probably) the most advanced open source database system
  - Free and open source
  - uses SQL (Structured query language) as its main query language
  - Some customers: Apple, Fujitsu, Red Hat, Cisco, Juniper Network
  - It has high availability.
  - It also supports image, video, audio storage and also supports graphical data.
  - It requires very low maintenance.
  - It supports Multi-version concurrency control (MVCC).
  - It has user defined data-types.
  - It runs on all operating systems.

SQL – PostgreSQL

- Download:
  - https://www.enterprisedb.com/downloads/postgres-postgresql-downloads
  - Version 16
  - Save your password!
  - Port: 5432
  - Save pre-installation summary:

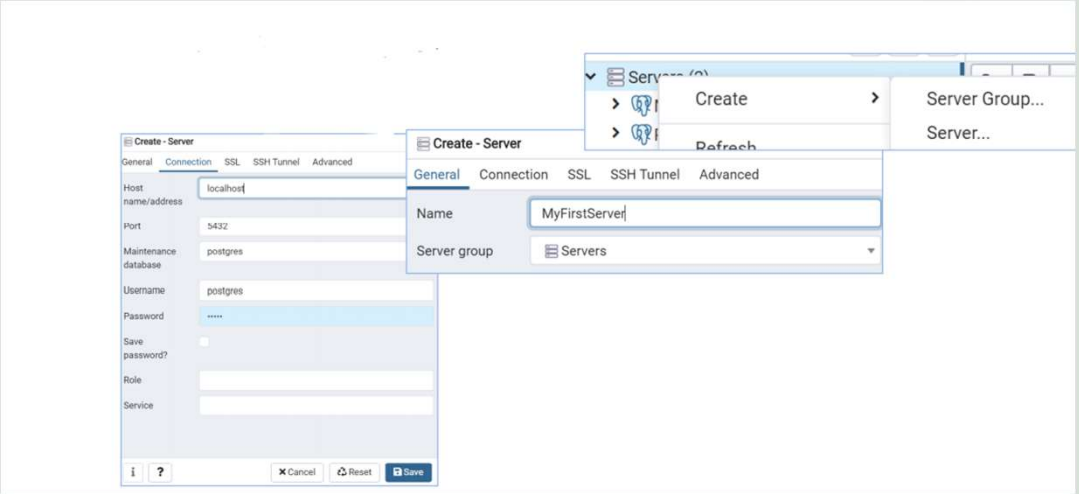


The following settings will be used for the installation::

```
Installation Directory: C:\Program Files\PostgreSQL\16
Server Installation Directory: C:\Program Files\PostgreSQL\16
Data Directory: C:\Program Files\PostgreSQL\16\data
Database Port: 5432
Database Superuser: postgres
Operating System Account: NT AUTHORITY\NetworkService
Database Service: postgresql-x64-16
Command Line Tools Installation Directory: C:\Program Files\PostgreSQL\16
pgAdmin4 Installation Directory: C:\Program Files\PostgreSQL\16\pgAdmin 4
Stack Builder Installation Directory: C:\Program Files\PostgreSQL\16
Installation Log: C:\Users\Utilizador\AppData\Local\Temp\install-postgresql.log
```

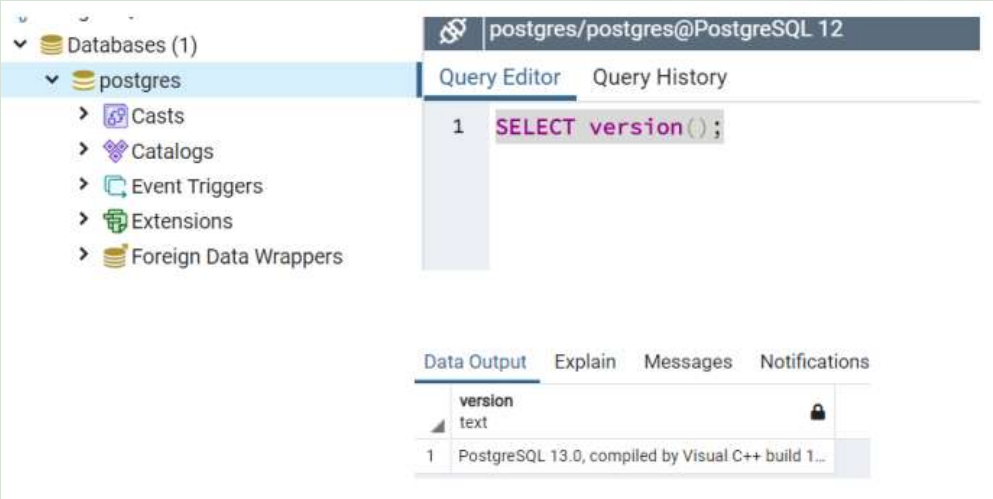
SQL – PostgreSQL

- Validate installation:
  - Open pgAdmin4.exe (c:\Programs\PostgreSQL\16\pgAdmin 4\runtime)
  - If no server exists, create one:
    - Right click on servers\create\server
    - Fill it



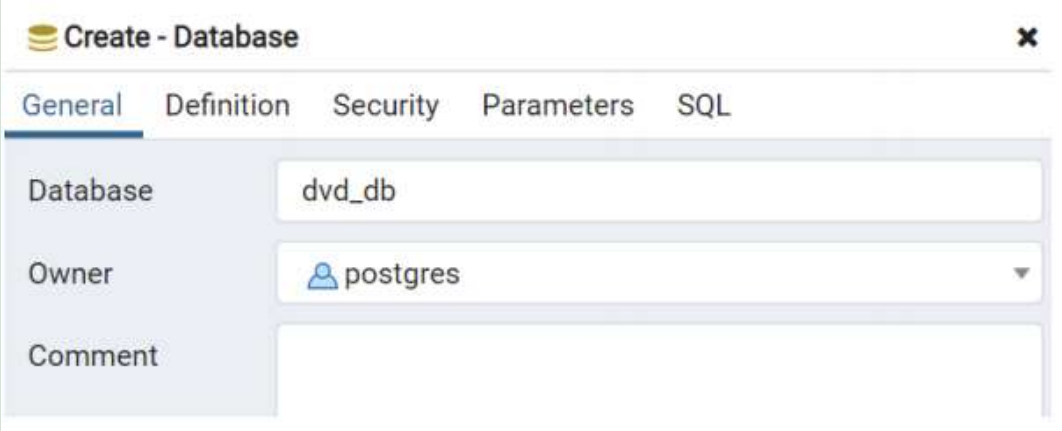
SQL – PostgreSQL

- Validate installation:
  - Open pgAdmin4 (c:\Programs\PostgreSQL 16\pgAdmin 4\bin
  - Right click on Database (1)\postgres -> Query Tool
  - Run query: select version()



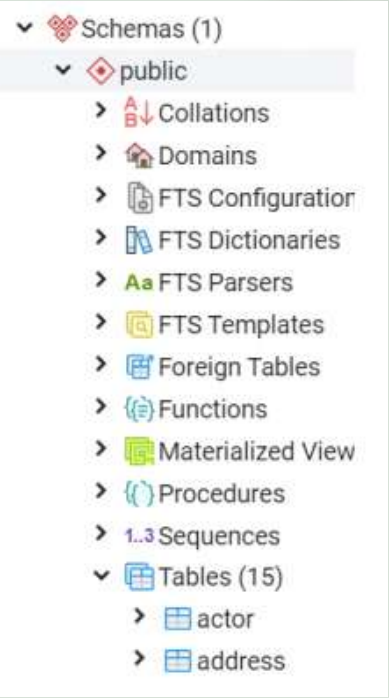
SQL – PostgreSQL

- Create database:
  - Download DVD Rental Sample Database
  - Right click on <server name>\Create\Database
  - Chose a name and save (eg. Dvd_db)
  - Right click on dvd_db\restore
  - On file name, insert fil path 'C:\Users\...\dvdrental.tar'
  - Click on Restore

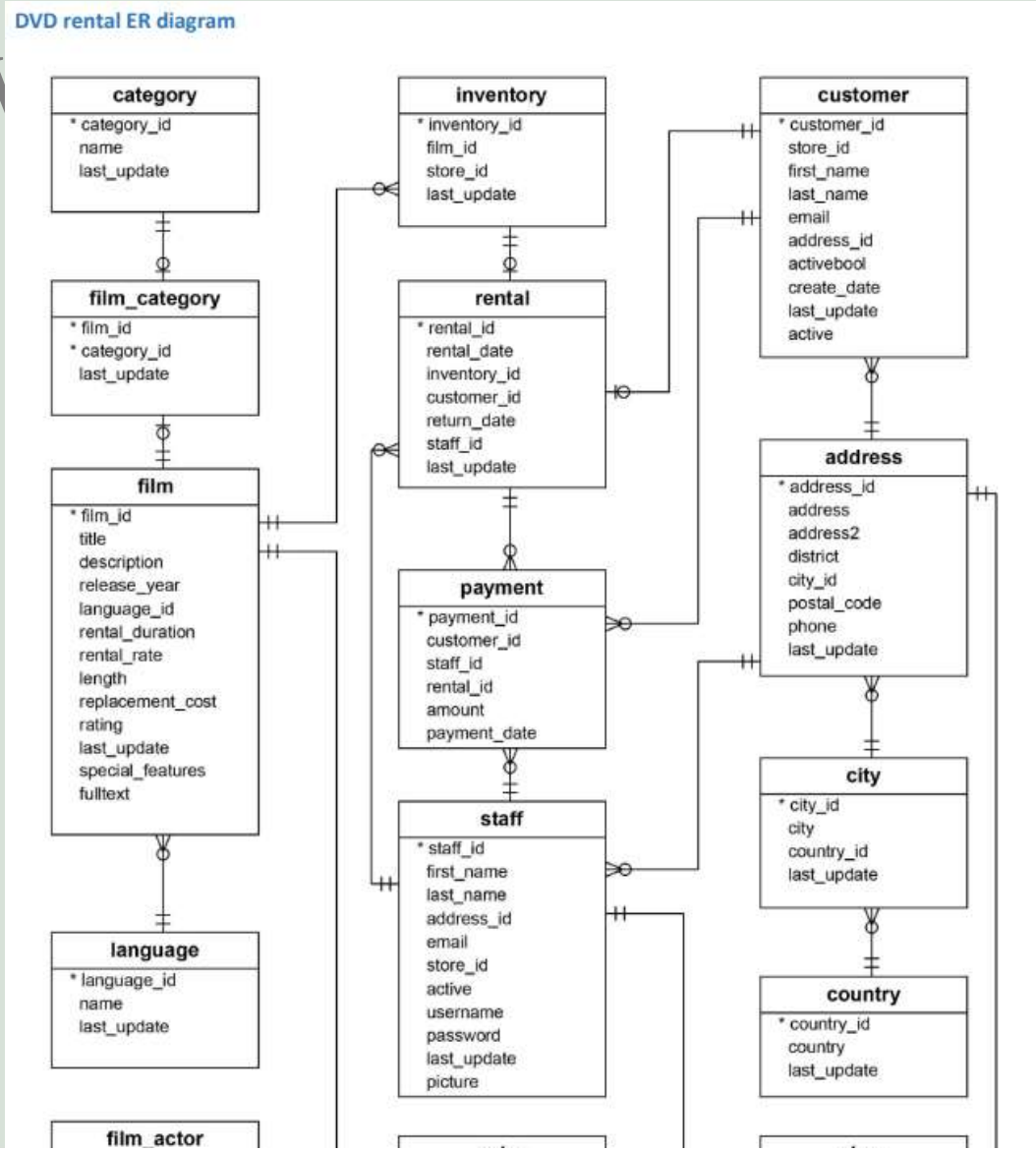


SQL – PostgreSQL

- Create database:
  - Confirm that you can see the tables
  - Right click in one of the tables\properties
  - Explore



SQL – DV ema


DVD rental ER diagram
category
* category_id
name
last_update
film_category
* film_id
* category_id
last_update
film
* film_id
title
description
release_year
language_id
rental_duration
rental_rate
length
replacement_cost
rating
last_update
special_features
fulltext
language
* language_id
name
last_update
film_actor
inventory
* inventory_id
film_id
store_id
last_update
rental
* rental_id
rental_date
inventory_id
customer_id
return_date
staff_id
last_update
payment
* payment_id
customer_id
staff_id
rental_id
amount
payment_date
staff
* staff_id
first_name
last_name
address_id
email
store_id
active
username
password
last_update
picture
customer
* customer_id
store_id
first_name
last_name
email
address_id
activebool
create_date
last_update
active
address
* address_id
address
address2
district
city_id
postal_code
phone
last_update
city
* city_id
city
country_id
last_update
country
* country_id
country
last_update

SQL – Select

From Actors,

Exercise:

1. Select only the first name ;
2. Select only first and last names;
3. Select only distinct first name;
4. Select only first name that is different from Nick;

Hint:

```
Select col1, col2
From table1
Where
<condition>;

/*
Comments
*/

/*    select * from
table1

means select all
data from table1
*/
```

SQL – Select

Exercise:

1. Select only the first name;
   Select first_name from actor;
2. Select only first and last names;
   Select first_name, last_name from actor;
3. Select only distinct first name;
   Select distinct first_name from actor;
4. Select only first name that is different from Nick;
   Select first_name, last_name
   From actor
   Where first_name != 'Nick';

Hint:

Select col1, col2
From table1
Where
<condition>;

/*
Comments
***/**

SQL – Order by

Exercise:

1. Select all actor's data sorted by last name;
2. Select all actor's data sorted by last name but descending;
3. Select all actor's data in which first name starts with 'B'

Hint:

```
Select col1, col2
From table1
Where <condition>
Order by col_1;
```

SQL – Order By

Exercise:

1. Select all actor's data sorted by last name;

   *select * from actor order by last_name;*

2. Select all actor's data sorted by last name but descending;

   *select * from actor order by last_name desc;*

3. Select all actor's data in which first name starts with 'B'

   *select * from actor where first_name like 'B%';*

Hint:

Select col1, col2
From table1
Where <condition>
Order by col_1;

SQL – Group by

Exercise:

1. Number of records of the actor's table;
2. How many actors are with the same first name sorted by first name descending;
3. Create an alias for the countings column (eg: howmany);

Hint:

Select col1, col2,…, count(colx)
From table1
Where <condition>
Group by col1, col2, …;

SQL – Group by

Exercise:

1. Number of records of the actor's table;

   Select count(*) from actor;

2. How many actors are with the same first_name;
3. Create an alias for the countings column (HowMany);

   select first_name, count(first_name) howmany

   from actor

   group by first_name

   order by first_name desc;

Hint:

Select col1, col2,…, count(colx)
From table1
Where <condition>
Group by col1, col2,…
Order by col1;

SQL – Group by

Exercise:

1. Reuse previous query and …
2. Sort it by HowMany in descending order and by first_name ascending;

Hint:

Select col1, col2,…, count(colx)
From table1
Where <condition>
Group by col1, col2,…
Order by col_1;

SQL – Group by

Exercise:

1. Reuse previous query and …
2. Sort it by HowMany in descending order and by first_name ascending;

   select first_name,count(first_name) howmany

   from actor

   group by first_name

   order by howmany desc, first_name;

Hint:

Select col1, col2,…, count(colx)
From table1
Where <condition>
Group by col_1, col2, …
Order by col_1;