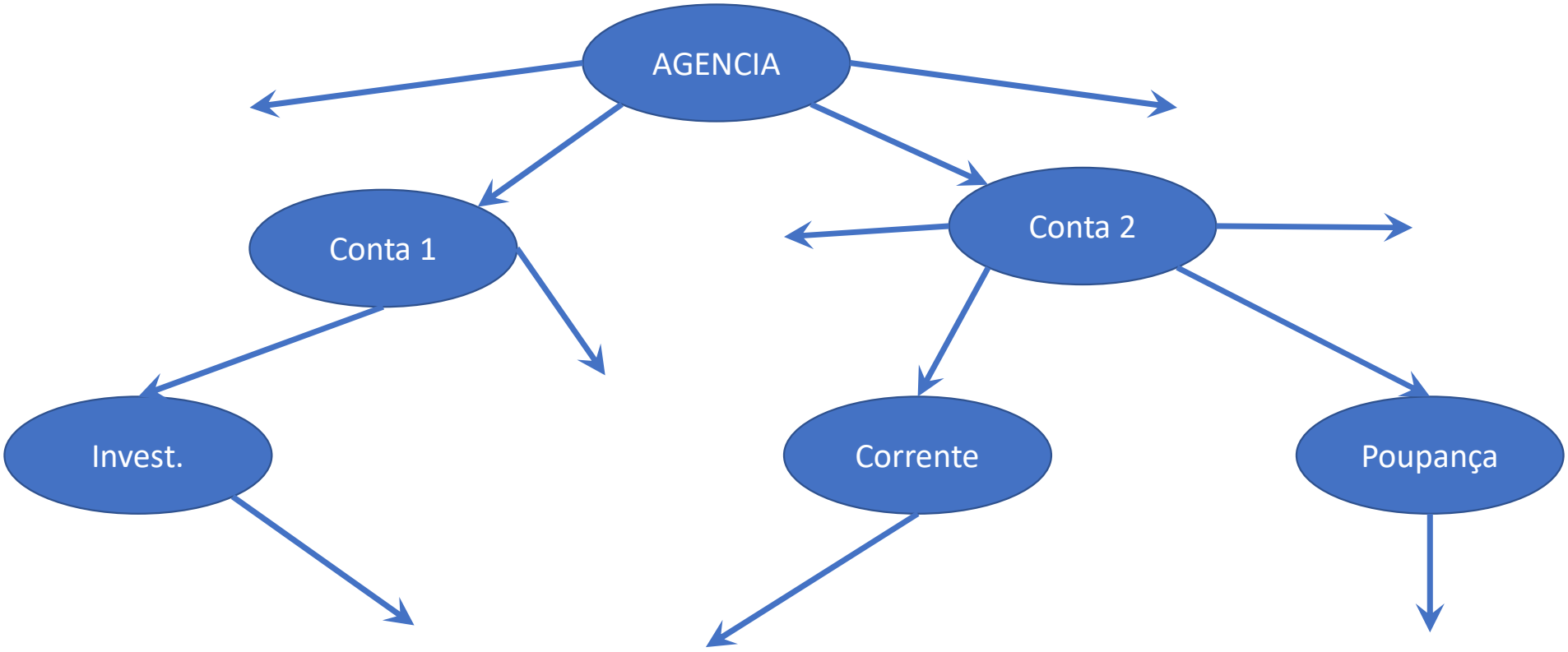
# Agenda da Aula

- Apresentação dos Alunos (Experiencia e Expectativa)
- Apresentação do Professor
- Apresentação do Plano de Ensino
  1. Critérios de Avaliação
  2. Critério de Registro de Frequência
  3. Formatos das Atividades do Curso
  4. Tópicos a serem abordados
  5. Processo de comunicação
- Orientação de TCC (Desenv. Web/Mobile, Cloud , RPA , BPM, AI, Machine Learning, etc.)
- Atividade de entrega (Esta tem Prazo de 1 semana)

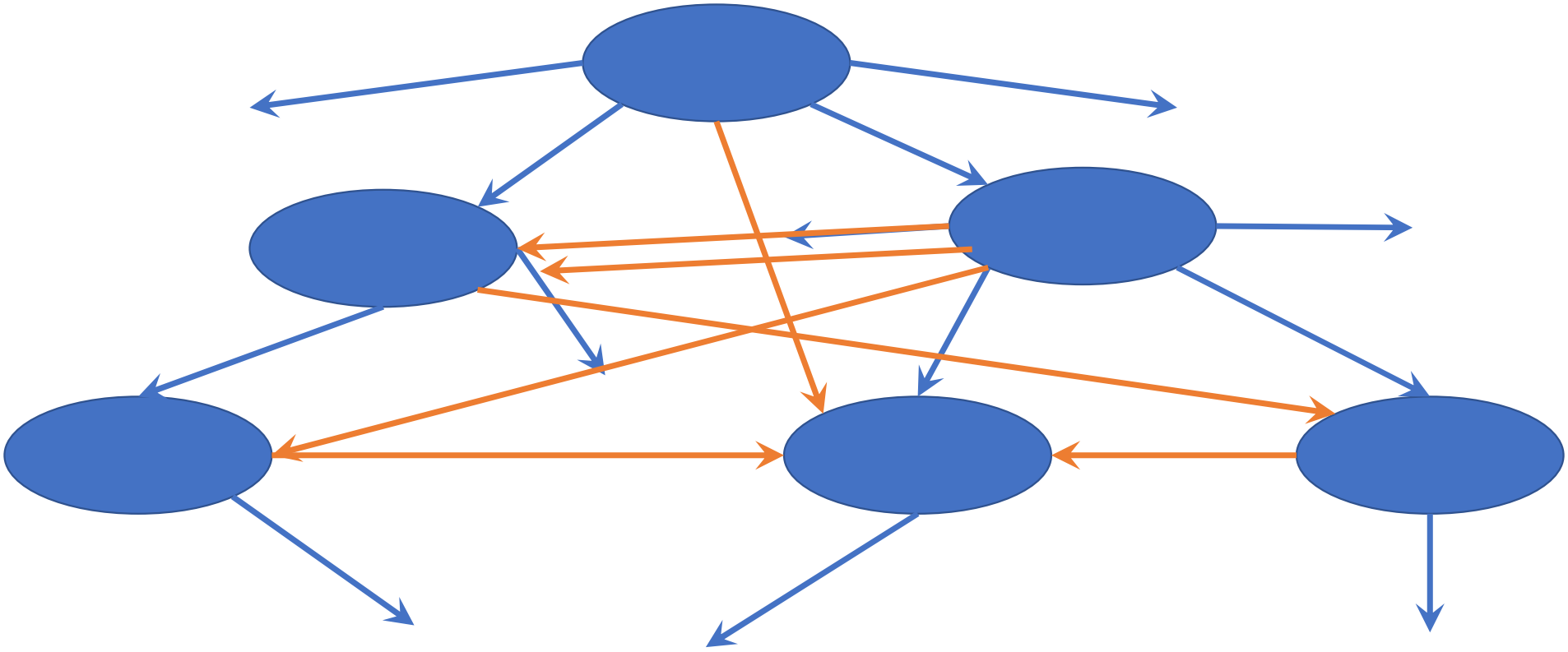
Prof. Joilson Cardoso

# Modelo Hierárquico



Prof. Joilson Cardoso

# Modelo Rede



Prof. Joilson Cardoso

# Modelo Relacional

Comparativo de jogo de Batalha Naval

| X/Y | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | ... |
|---|---|---|---|---|---|---|---|---|---|
| 1 | ? | | | | | | | | |
| 2 | | | | | ? | | | | |
| 4 | | | ? | | | | | | |
| 5 | | | | | | | ? | | |
| 6 | | | | | | | | | |
| 7 | | | | | ? | | | | |
| ... | | | | | | | | | |

- **Teoria dos Conjuntos** - Álgebra Relacional ( Segmento da Matemática) – Conceito de Tuplas (x,y)
- Traduzido no Modelo Relacional – como Linhas/Colunas
- 1970 – E. Codd – Criou o SQL (STRUCTURED QUERY LANGUAGE)

Fonte: Sistema de Banco de Dados (Silberschatz, Abraham)

Prof. Joilson Cardoso

# Estrutura Básica

TABELA (LINHAS E COLUNAS)

| Linhas/Colunas | *Coluna 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | ... |
|---|---|---|---|---|---|---|---|---|---|
| Linha 1 | 1 | Joao | Rua X | 01/12/2022 | ... | ... | ... | ... | ... |
| 2 | 2 | Pedro | Rua Y | 13/04/2016 | ... | ... | ... | ... | ... |
| 3 | 3 | Carmelo | Rua Z | 20/03/2008 | ... | ... | ... | ... | ... |
| 4 | 4 | Jacinta | Rua A | | ... | ... | ... | ... | ... |
| 5 | 5 | Maria | Rua B | 04/07/1999 | ... | ... | ... | ... | ... |
| 6 | 6 | Eugenia | Rua C | | ... | ... | ... | ... | ... |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |

- ***Coluna 1** - Neste EXEMPLO pode ser a CHAVE PRIMARIA. Conceito: NÃO SE REPETE E SEMPRE TEM VALOR

Prof. Joilson Cardoso

# Revisão SQL Básico

- Comandos DDL (Data Definition Language)
- Comandos DML (Data Manipulation Language)
- Atividade de entrega (Prazo até as 22:50)

Prof. Joilson Cardoso

# Revisão SQL Básico

- Comandos DDL (Data Definition Language)

CREATE TABLE *table_name* (
*column1 datatype constraint*,
*column2 datatype constraint*,
*column3 datatype constraint*,
....
);

```
CREATE TABLE Persons (
    PersonID int,
    LastName varchar(255) not null,
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255),
    Birth Datetime
);
```

```
ALTER TABLE Persons add
CONSTRAINT pkPersons Primary Key
(PersonID);
```

Prof. Joilson Cardoso

# Revisão SQL Básico

- Comandos DDL (Data Definition Language)

## DATATYPES

- **CHAR(n)**- Caracter fixo de **n** bytes (n até 255)
- **VARCHAR(n)** - Caracter variavel de n bytes (n depende do SGBD)
- **DECIMAL (p, n)** – Numerico que ocupa **8 bytes** (p – precisão, n - escala)
- **LONG** - Numerico que ocupa **8 bytes** no disco
- **INT**- Numerico que ocupa **4 bytes** no disco
- **SMALLINT** - Numerico que ocupa **2 bytes** no disco
- **BOOLEAN /BIT** – Numerico que ocupa **1 bit** no disco
- **DATE /DATETIME / TIME** - Numerico que ocupa **8 bytes** (**depende do SGBD**)
- **BLOBs**- Binary Large Objects (Binary, Image, etc. Normalmente **até 2GB**)
- **CLOBS** – Character Large Objects (LongVarchar, Text, etc. Normalmente **até 2GB**)

OBS: Tem SGBDs que usam nomes específicos , como: **NUMERIC(p,n)** (MSSqlServer) , **NUMBER(p,n)** (Oracle) , etc. No entanto TODOS eles aceitam os tipos do PADRAO ANSI.

Prof. Joilson Cardoso

# Revisão SQL Básico

- Comandos DDL (Data Definition Language)

## TIPOS DE CONSTRAINTS (Restrições)

- **NOT NULL** - Garante que uma coluna nao tenha NULL
- **UNIQUE** - Garante que TODOS os valores sejam diferentes
- **PRIMARY KEY** - Combinação de NOT NULL com UNIQUE. Pode ser referenciado por outra tabela.
- **FOREIGN KEY** - Identifica valor da coluna em outra tabela.
- **CHECK** - Verifica se o valor aplica a determinada condição
- **DEFAULT** - Coloca um valor padrão, quando nao especificado.
- **INDEX** - Cria um índice para a referida coluna

# Revisão SQL Básico

- Comandos DDL (Data Definition Language)

CRIAÇÃO DE CONSTRAINTS FOREIGN KEY

```
ALTER TABLE <tabela FILHA>
ADD CONSTRAINT <nome constraint>
FOREIGN KEY (nome coluna) REFERENCES <tabela PAI>
 (<opcional nome da coluna na tabela pai>);

ALTER TABLE Orders
ADD CONSTRAINT FK_PersonOrder
FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);
```

# Revisão SQL Básico

- Comandos DDL (Data Definition Language) – Outra forma de criar Tabela.

```
CREATE TABLE new_table_name AS
    SELECT column1, column2,...
    FROM existing_table_name
    WHERE ....;
```

```
CREATE TABLE TestTable AS
SELECT customername,  contactname
FROM customers ;
```

# Revisão SQL Básico

- Comandos DML (Data Manipulation Language) - INSERT

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

```
INSERT INTO Customers (CustomerName, ContactName,
Address, City, PostalCode, Country)
VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen
21', 'Stavanger', 4006, 'Norway');
```

**OBS: NÃO CONFIE NA ORDEM QUE O SGBD VAI CRIAR AS COLUNAS. SEMPRE DESCREVA A ORDEM DAS COLUNAS.**

# Revisão SQL Básico

- Comandos DML (Data Manipulation Language) - UPDATE

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

```
UPDATE Customers
SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
WHERE CustomerID = 1;
```

**OBS: NUNCA ..NUNCA ...NUNCA ALTERE UMA COLUNA CHAVE PRIMÁRIA.**

# Revisão SQL Básico

- Comandos DML (Data Manipulation Language) - DELETE

DELETE FROM *table_name* WHERE *condition*;

DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';
(POSSIVEL , MAS EVITE)

DELETE FROM Customers WHERE CustomerID= 1450;
(IDEAL, PREFIRA USAR CHAVE PRIMARIA )

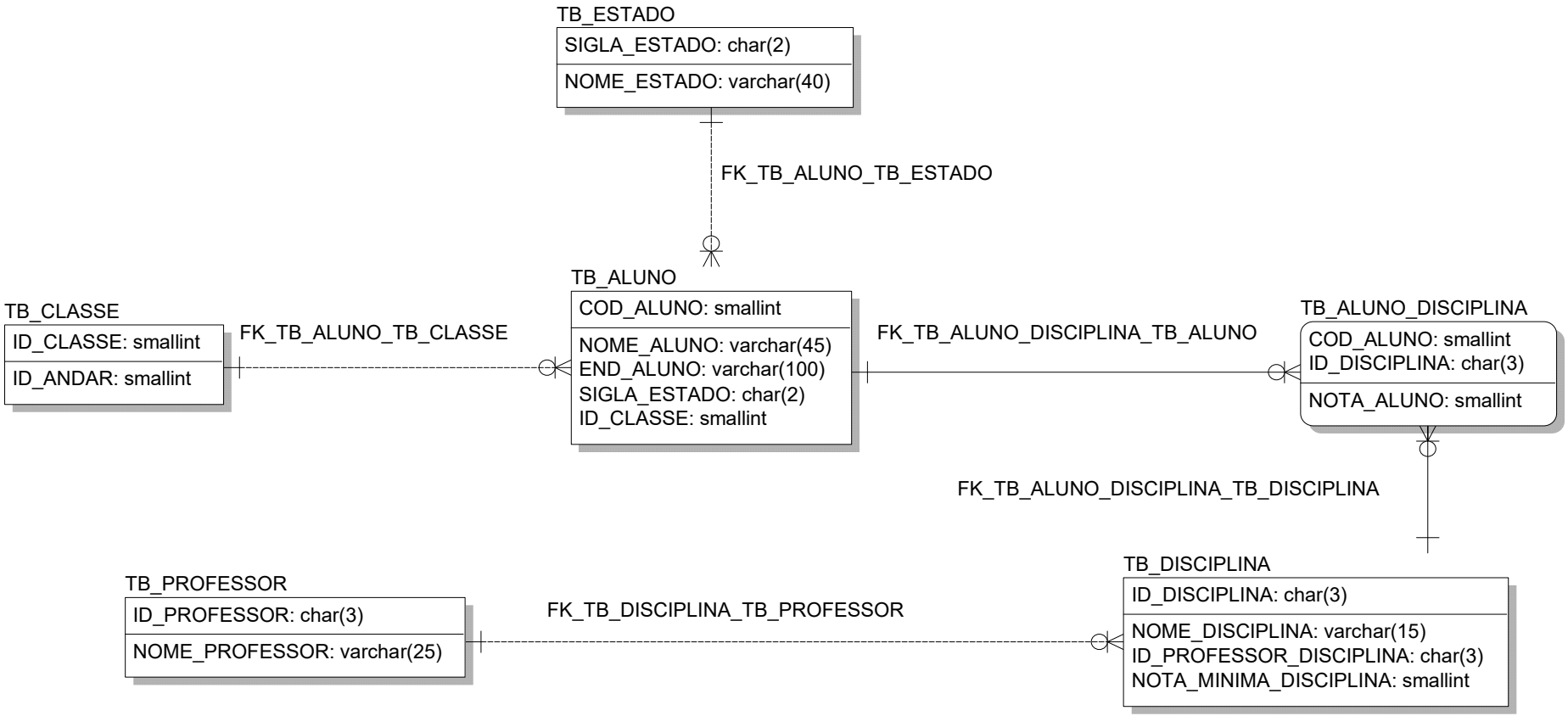
**OBS: NUNCA ..NUNCA ...NUNCA ESQUEÇA DA CLAUSULA WHERE. Caso tenha que eliminar TODOS os registros, prefira o comando TRUNCATE TABLE.**

FATEC ZL - Laboratório de Banco de Dados

Prof. Joilson Cardoso

# Revisão SQL Básico

- MODELO DE DADOS - ALUNO



Prof. Joilson Cardoso

# Revisão SQL Básico

- ATIVIDADE (Tarefa do Teams)

Prof. Joilson Cardoso

# Revisão SQL Básico

- Comando SELECT
- Funções de Agregação
- Atividade de entrega (Prazo até as 22:50)

Prof. Joilson Cardoso

# Revisão SQL Básico

- Comandos DML (Data Manipulation Language) – SELECT

```
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

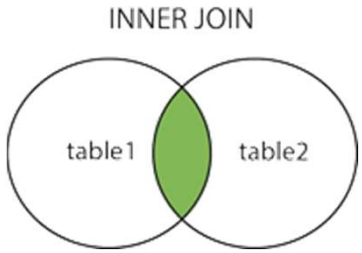
```
SELECT name FROM Customers
WHERE Country='Mexico';
```

**OBS: UTILIZE a tag ( * ), APENAS em IDE de Desenvolvimento.**

# Revisão SQL Básico

- Comandos DML– SELECT com INNER JOIN

```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name;
```




```
SELECT Orders.OrderID, Customers.CustomerName
FROM Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

**OBS: Nome da Tabela + "." + Nome da Coluna → identifica o dado a ser buscado ou comparado. Este nome da Tabela com o "." é opcional quando os nomes das colunas envolvidas na query são DISTINTOS.**

**RECOMENDÁVEL SEMPRE utilizar ALIAS. A query fica mais LEGÍVEL.**

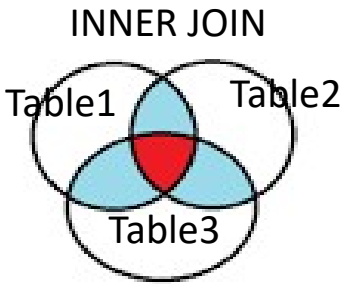
Prof. Joilson Cardoso

# Revisão SQL Básico

- Comandos DML– SELECT com INNER JOIN com CORRELAÇÃO

```
SELECT column_name(s)
FROM table1 <alias1>
INNER JOIN table2 <alias2>
ON (<alias1>.column_name = <alias2>.column_name)
INNER JOIN table3 <alias3>
ON (<alias2>.column_name = <alias3>.column_name);
```

```
SELECT A.OrderID, B.CustomerName
FROM Orders A
INNER JOIN Customers B
ON A.CustomerID = B.CustomerID
INNER JOIN Shippers  C
ON A.ShipperID = C.ShipperID;
```

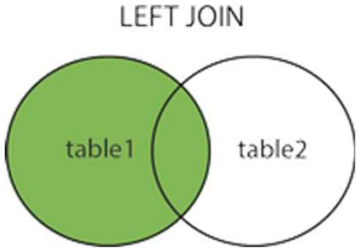



**OBS: O Resultado do INNER JOIN é a INTERSECÇÃO dos valores no conjunto Resultado.**

# Revisão SQL Básico

- Comandos DML– SELECT com LEFT OUTER JOIN (LEFT JOIN)

```
SELECT column_name(s)
FROM table1 <alias1>
LEFT OUTER JOIN table2 <alias2>
ON (<alias1>.column_name = <alias2>.column_name);
```




```
SELECT Orders.OrderID, Customers.CustomerName
FROM Orders A
LEFT JOIN Customers B
ON A.CustomerID = B.CustomerID;
```

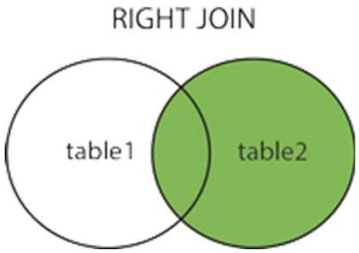
**OBS: O Resultado do LEFT JOIN é a INTERSECÇÃO dos valores no conjunto Resultado + o resultado da Tabela definida a esquerda na SINTAXE , mesmo que a JUNCAO tenha NULOs na tabela da ESQUERDA .**

Prof. Joilson Cardoso

# Revisão SQL Básico

- Comandos DML– SELECT com RIGHT OUTER JOIN (RIGHT JOIN)

```
SELECT column_name(s)
FROM table1 <alias1>
RIGHT OUTER JOIN table2 <alias2>
ON (<alias1>.column_name = <alias2>.column_name);
```



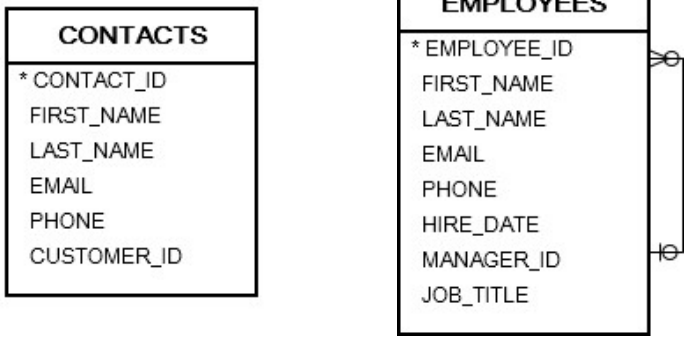

```
SELECT Orders.OrderID, Customers.CustomerName
FROM Orders A
RIGHT  JOIN Customers B
ON A.CustomerID = B.CustomerID;
```

**OBS: O Resultado do RIGHT JOIN é a INTERSECÇÃO dos valores no conjunto Resultado + o resultado da Tabela definida a direita na SINTAXE , mesmo que a JUNCAO tenha NULOs na tabela da DIREITA .**

Prof. Joilson Cardoso

# Revisão SQL Básico

- Comandos DML– UNION / UNION ALL

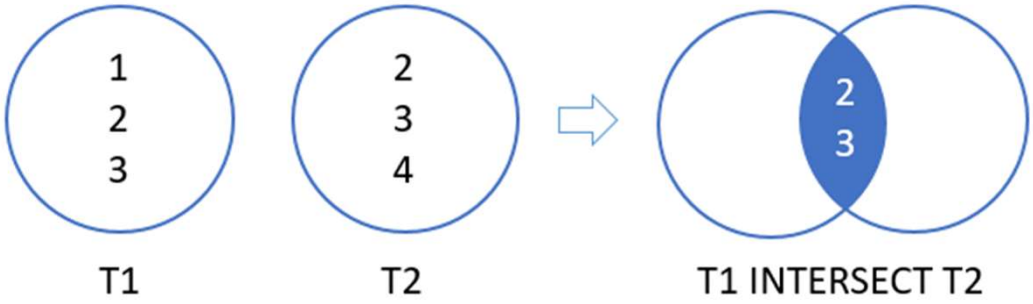

```
SELECT first_name, last_name, email, 'contact'
FROM contacts
UNION
SELECT first_name, last_name, email, 'employee'
FROM employees;
```


**OBS: UNION faz um DISTINCT (Elimina registros duplicados), enquanto o UNION ALL traz TODOS os registros, inclusive duplicados.**

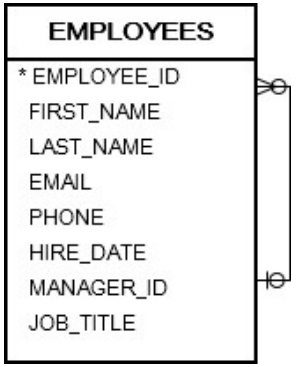
Prof. Joilson Cardoso

# Revisão SQL Básico

- Comandos DML– INTERSECT







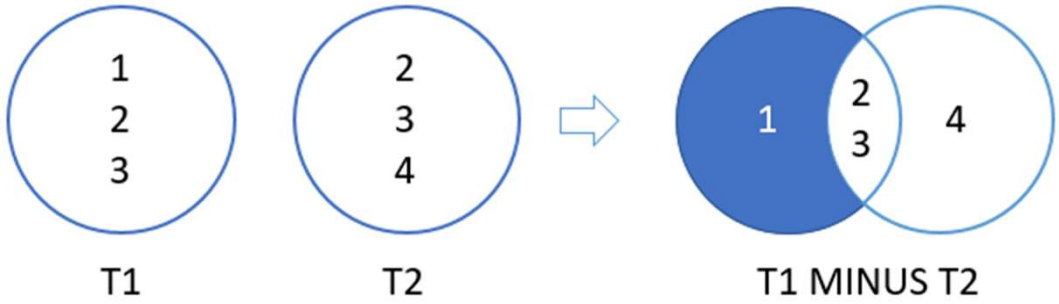

**SELECT** last_name **FROM** contacts
**INTERSECT**
**SELECT** last_name **FROM** employees
**ORDER BY** last_name;

**OBS: Tem SGBD que NÃO SUPORTA o INTERSECT (MySql suporta a partir da versão 8.0.31, por exemplo). A solução é fazer usando SUBSELECT.**

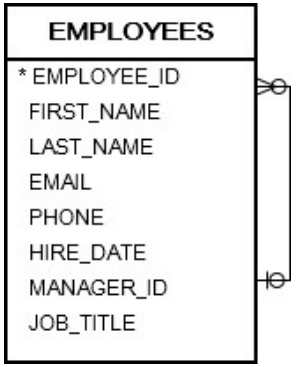
Prof. Joilson Cardoso

# Revisão SQL Básico

- Comandos DML– MINUS / EXCEPT








**SELECT** last_name **FROM** contacts
**MINUS**
**SELECT** last_name **FROM** employees
**ORDER BY** last_name;

**OBS: Tem SGBD que NÃO SUPORTA o MINUS / EXCEPT (MySql suporta a partir da versão 8.0.31, por exemplo). A solução é fazer usando LEFT JOIN.**

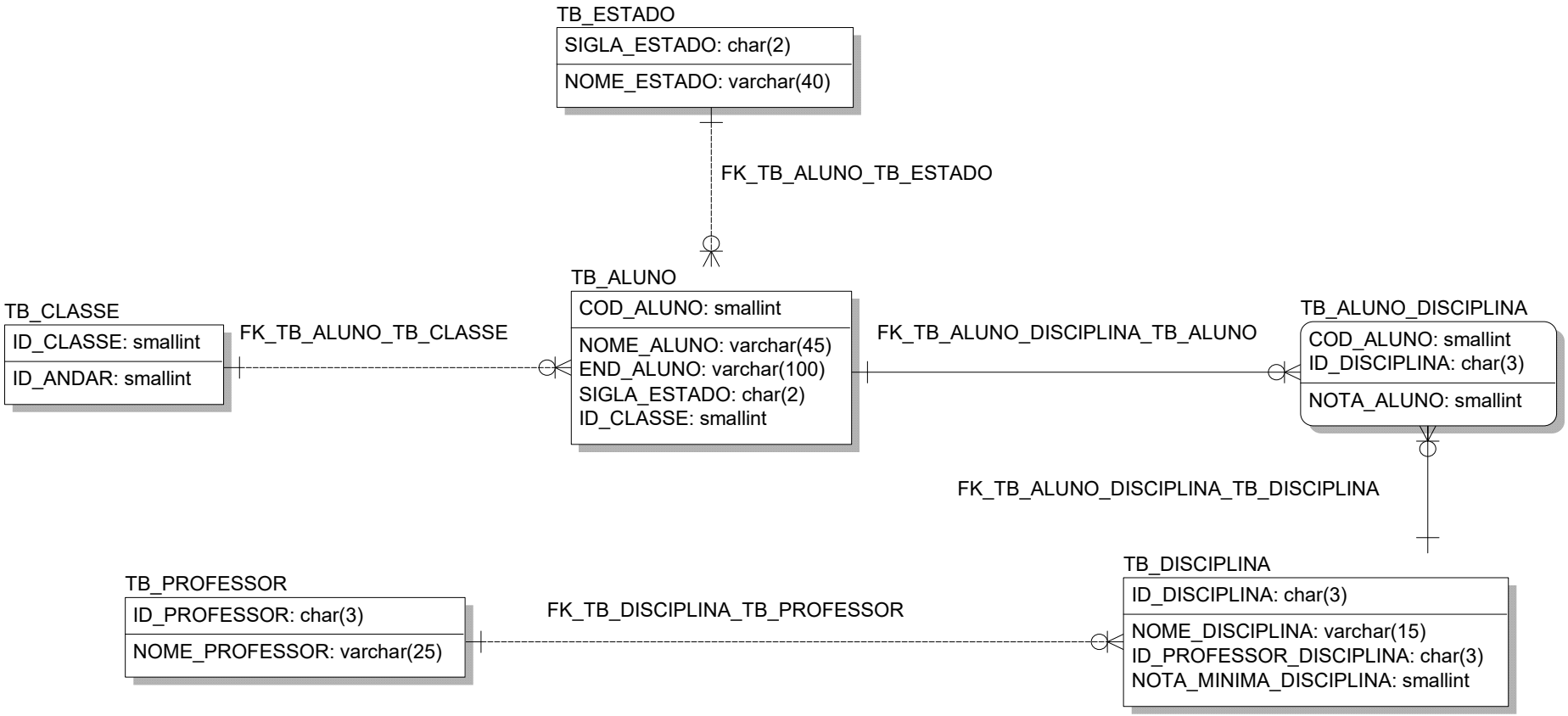
**SELECT A.last_name FROM** contacts A
**LEFT JOIN FROM** employees B
ON CUSTOMER_ID = EMPLOYEE_ID
**WHERE B.**EMPLOYEE_ID **IS NULL**;

FATEC ZL - Laboratório de Banco de Dados

Prof. Joilson Cardoso

# Revisão SQL Básico

- MODELO DE DADOS - ALUNO



Prof. Joilson Cardoso

# Revisão SQL Básico

- ATIVIDADE (Tarefa do Teams)

Prof. Joilson Cardoso

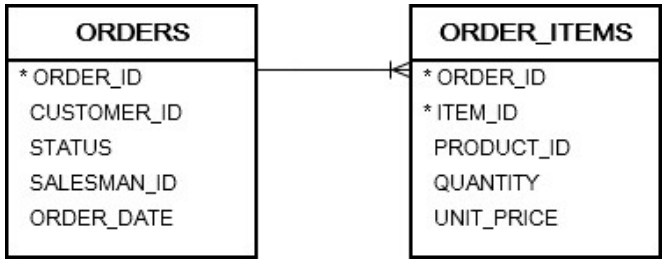
# Revisão SQL Básico

- FUNCOES AGREGADAS

**MIN()**
**MAX()**
**AVG()**
**COUNT()**
**SUM()**

**OBS:**

1. **QUANDO UTILIZADAS EM CONJUNTO COM OUTRAS COLUNAS , DEVE VIR ACOMPANHADA DA CLAUSULA GROUP BY, PARA TODAS AS COLUNAS QUE ESTEJA ACOMPANHANDO . EVENTUALMENTE PODE-SE TAMBEM UTILIZAR A CLAUSULA HAVING.**
2. **EVITE UTILIZAR "*" com FUNCOES AGREGADAS dentro dos Programas.**




```
SELECT customer_id, COUNT( order_id )
FROM orders
GROUP BY customer_id ;
```

Prof. Joilson Cardoso

# Revisão SQL Básico

- ATIVIDADE (Tarefa do Teams)

Prof. Joilson Cardoso

# Stored Procedure

- Conceitos
- Exemplos
- Atividade de entrega (Prazo até as 22:50)

FATEC ZL - Laboratório de Banco de Dados

Prof. Joilson Cardoso

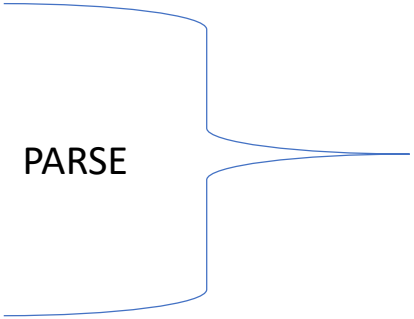
# Stored Procedure

## Passos para Execução de comando SQL

SINTAXE

PERMISSÕES

PLANO DE EXECUÇÃO.




COMPILAÇÃO ( +/- 75% do tempo)

EXECUÇÃO

EXECUÇÃO (+/- 25% do tempo)

Prof. Joilson Cardoso

# Stored Procedure

Conceito – Procedimento em linguagem de programação estruturada previamente COMPILADO e ARMAZENADO no Dicionário de Dados do SGBD.

SINTAXE no MySQL:

CREATE [DEFINER = *user*] PROCEDURE *sp_name* ([*proc_parameter*[,...]])
*routine_body*

*proc_parameter*: [ IN | OUT | INOUT ] *param_name type*

*routine_body*: *Comandos Validos da Linguagem SQL*

Prof. Joilson Cardoso

# Stored Procedure

## Exemplo Mysql:

```
Mysql> delimiter //
mysql> CREATE PROCEDURE citycount (IN country CHAR(3), OUT cities INT)
BEGIN
SELECT COUNT(country) INTO cities FROM world.city
                        WHERE CountryCode = country;
END//
mysql> delimiter ;

mysql> CALL citycount('JPN', @cities); -- cities in Japan

mysql> SELECT @cities;
```

Prof. Joilson Cardoso

# Stored Procedure

## Exemplo Oracle:

```
CREATE OR REPLACE PROCEDURE add_new_supplier
  (supplier_id_in IN NUMBER, supplier_name_in IN VARCHAR2)
IS
BEGIN
  INSERT INTO suppliers (supplier_id, supplier_name )
                  VALUES ( supplier_id_in, supplier_name_in );
END;
```

Prof. Joilson Cardoso

# Stored Procedure

- ATIVIDADE (Tarefa do Teams)