Banco de Dados II

Prof. Vinícius Alves Hax

Na aula de hoje

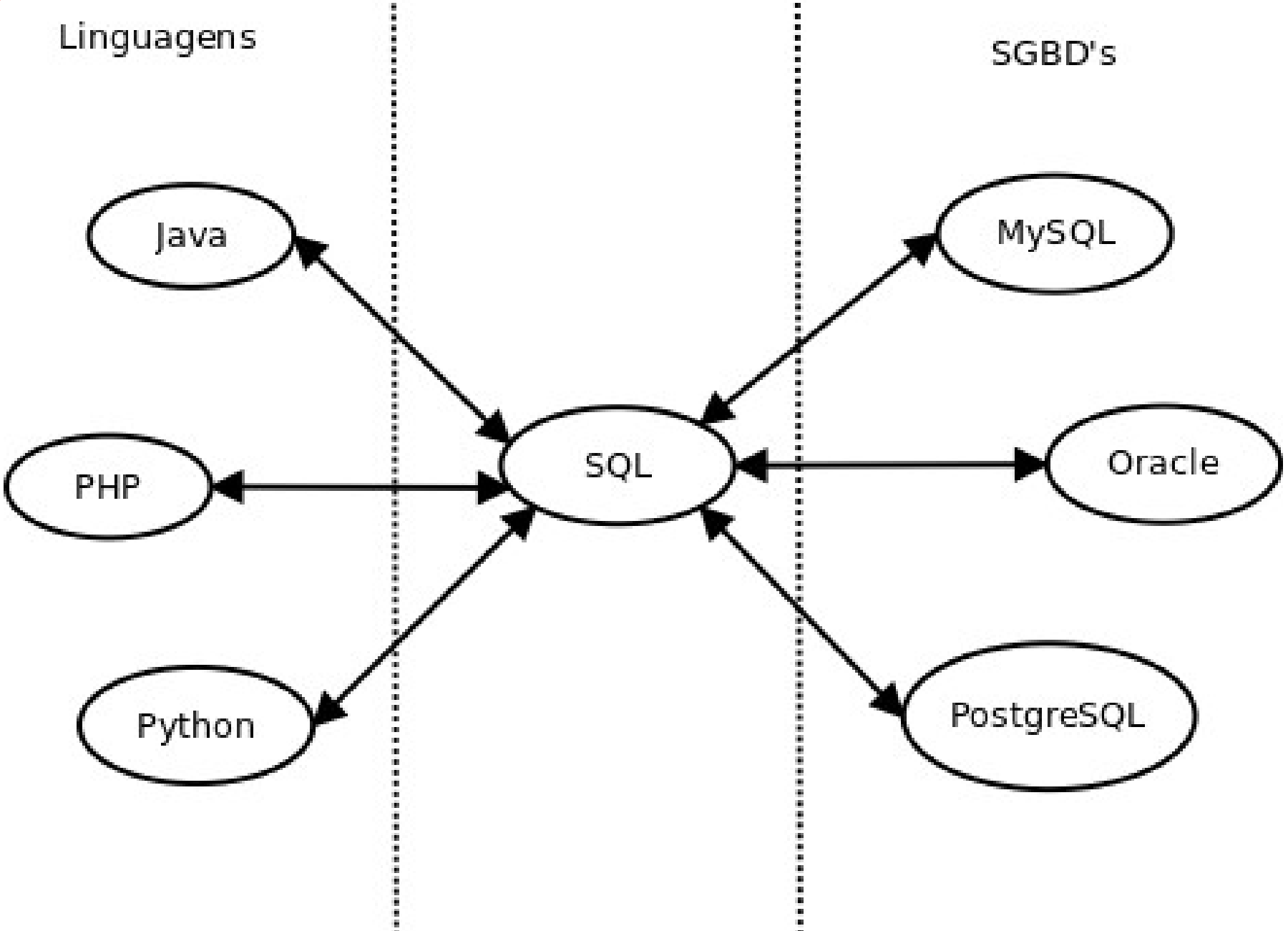
- Revisão sobre SQL

Para que existe o SQL?

Para que existe o SQL?

R: Padronizar a comunicação com o banco de dados facilitando o desenvolvimento de software. Na prática também facilita a troca de tecnologias.

Importância do SQL



Subconjuntos de SQL

- DDL (Data Definition Language): CREATE, DROP e ALTER
- DML (Data Manipulation Language): INSERT, UPDATE, DELETE
- DCL (Data Control Language): GRANT, REVOKE
- DTL (Data Transaction Language): BEGIN WOK, COMMIT, ROLLBACK
- DQL (Data Query Language): SELECT

CREATE

- CREATE DATABASE nome;
- CREATE TABLE table_name(

  column1 datatype,

  column2 datatype,

  column3 datatype,

  .....

  columnN datatype,

  PRIMARY KEY(one or more columns)

);

INSERT

- INSERT INTO TABLE_NAME (column1, column2, column3,...columnN) VALUES (value1, value2, value3,...valueN);

SELECT

- SELECT column1, column2, columnN FROM table_name;
- SELECT column1, column2, columnN

  FROM table_name

  WHERE [search_condition]

UPDATE

- UPDATE table_name SET column1 = value1, column2 = value2...., columnN = valueN WHERE [condition];

  (sem o WHERE atualiza todos os registros)

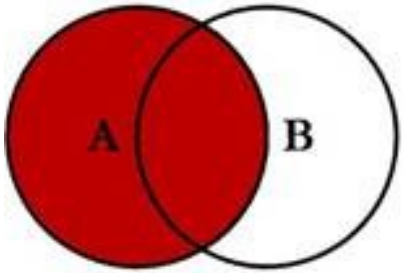
DELETE

- DELETE FROM table_name WHERE [condition];

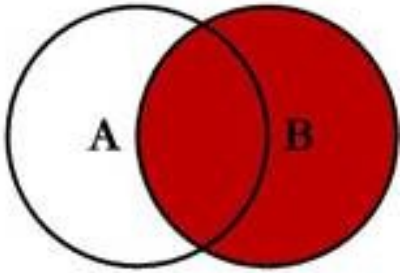
  (sem o WHERE deleta todos os registros!!!)

Joins = Junção

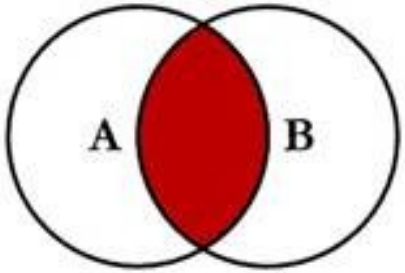
SQL JOINS



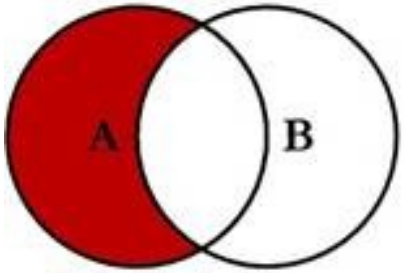

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



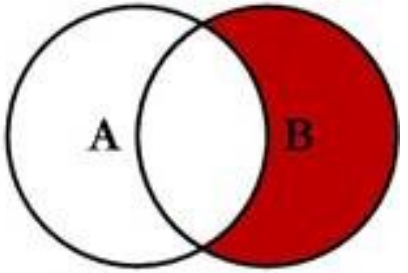

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



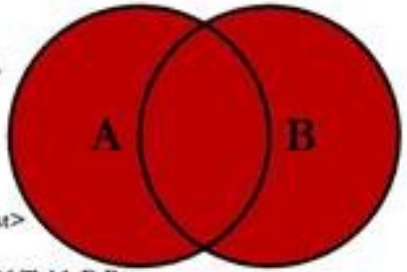

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



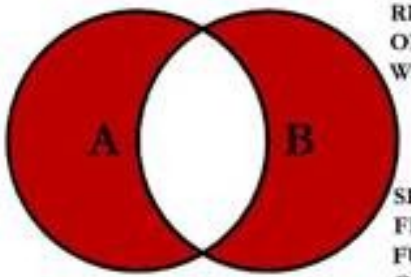

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

© C.L. Moffatt, 2008

Operação Cross Join

Também chamada de junção cruzada ou ainda de produto cartesiano

Dado:

Tabela A

ColunaA
1
2

Tabela B

ColunaB
A
B
C

A X B =

ColunaA	ColunaB
1	A
1	B
1	C
2	A
2	B
2	C

Praticando

- Baixe o arquivo de dados em:

  https://encurtador.com.br/HuDMi

  ou

  https://github.com/viniciusalveshax/aulas-2024/blob/master/banco-de-dados-2/aula3/exemplo_join.sql (se o endereço acima não funcionar)

INNER JOIN

- Mostra somente o que está referenciado em ambas as tabelas:
- SELECT t.name as teacher_name, s.name as student_name

  FROM teacher as t

  INNER JOIN student as s

  ON t.id = s.teacher_id;

Query | Query History

```
1 SELECT t.name as teacher_name, s.name as student_name
2 FROM teacher as t
3 INNER JOIN student as s
4 ON t.id = s.teacher_id;
```

Data Output | Messages | Notifications

SQL

	teacher_name character varying (200)	student_name character varying (200)
1	Yoda	Luke
2	Obi Wan	Anakin
3	Palpatine	Anakin
4	Qui-gon Jin	Obi Wan
5	Dookan	Qui-gon Jin
6	Anakin	Ahsoka
7	Luke	Rei
8	Luke	Kylo Ren

LEFT JOIN

- Mostra os dados com base na tabela da esquerda:
- SELECT t.name as teacher_name, s.name as student_name

  FROM teacher as t

  LEFT JOIN student as s

  ON t.id = s.teacher_id;

Query | Query History

```
1 SELECT t.name as teacher_name, s.name as student_name
2 FROM teacher as t
3 LEFT JOIN student as s
4 ON t.id = s.teacher_id;
```

Data Output | Messages | Notifications

SQL

	teacher_name character varying (200)	student_name character varying (200)
1	Yoda	Luke
2	Obi Wan	Anakin
3	Palpatine	Anakin
4	Qui-gon Jin	Obi Wan
5	Dookan	Qui-gon Jin
6	Anakin	Ahsoka
7	Luke	Rei
8	Luke	Kylo Ren
9	Mace Windu	[null]

RIGHT JOIN

- Mostra os dados com base na tabela da direita:
- SELECT t.name as teacher_name, s.name as student_name

  FROM teacher as t

  RIGHT JOIN student as s

  ON t.id = s.teacher_id;

Query | Query History

```
1  SELECT t.name as teacher_name, s.name as student_name
2  FROM teacher as t
3  RIGHT JOIN student as s
4  ON t.id = s.teacher_id;
```

Data Output | Messages | Notifications

SQL

	teacher_name character varying (200)	student_name character varying (200)
1	Yoda	Luke
2	Obi Wan	Anakin
3	Palpatine	Anakin
4	Qui-gon Jin	Obi Wan
5	Dookan	Qui-gon Jin
6	Anakin	Ahsoka
7	Luke	Rei
8	Luke	Kylo Ren
9	[null]	Palpatine

FULL JOIN

- Mostra os dados com base nas duas tabelas:
- SELECT t.name as teacher_name, s.name as student_name

  FROM teacher as t

  FULL JOIN student as s

  ON t.id = s.teacher_id;

Query | Query History

```
1  SELECT t.name as teacher_name, s.name as student_name
2  FROM teacher as t
3  FULL JOIN student as s
4  ON t.id = s.teacher_id;
```

Data Output | Messages | Notifications

	teacher_name character varying (200)	student_name character varying (200)
1	Yoda	Luke
2	Obi Wan	Anakin
3	Palpatine	Anakin
4	Qui-gon Jin	Obi Wan
5	Dookan	Qui-gon Jin
6	Anakin	Ahsoka
7	Luke	Rei
8	Luke	Kylo Ren
9	[null]	Palpatine
10	Mace Windu	[null]

CROSS JOIN

- Combina todas as possibilidades das duas tabelas
- SELECT t.name as teacher_name, s.name as student_name

  FROM teacher as t

  CROSS JOIN student as s;

Query | Query History

```
1 SELECT t.name as teacher_name, s.name as student_name
2 FROM teacher as t
3 CROSS JOIN student as s;
```

Data Output | Messages | Notifications

	teacher_name character varying (200)	student_name character varying (200)
1	Palpatine	Luke
2	Yoda	Luke
3	Dookan	Luke
4	Qui-gon Jin	Luke
5	Obi Wan	Luke
6	Anakin	Luke
7	Luke	Luke
8	Mace Windu	Luke
9	Palpatine	Anakin
10	Yoda	Anakin
11	Dookan	Anakin
12	Qui-gon Jin	Anakin
13	Obi Wan	Anakin
14	Anakin	Anakin
15	Luke	Anakin

Referências

- https://pt.wikipedia.org/wiki/SQL
- https://www.tutorialspoint.com/postgresql/postgresql_delete_query.htm
- https://www.devmedia.com.br/sql-join-entenda-como-funciona-o-retorno-dos-dados/31006