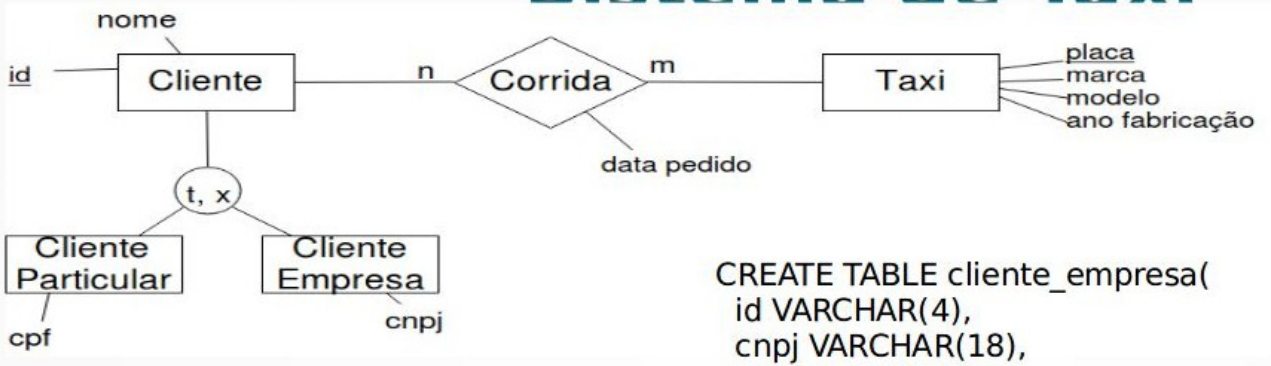
Banco de dados SQL

- Joins e Views

Esquema adotado

- O esquema utilizado nos exemplos é o mesmo do conjunto de slides anterior:
  - Introdução a SQL de Fagner Pantoja (baseado em André Santachè e Jaudete Daltio)
- Veja esquema no slide seguinte

Sistema de Taxi



Script para criar banco sistema_taxi

```
CREATE DATABASE sistema_taxi
USE sistema_taxi

CREATE TABLE cliente(
  id VARCHAR(4),
  nome VARCHAR(80),
  PRIMARY KEY(id)
);
CREATE TABLE cliente_particular(
  id VARCHAR(4),
  cpf VARCHAR(14),
  PRIMARY KEY(id),
  FOREIGN KEY(id) REFERENCES cliente(id)
);
```

```
CREATE TABLE cliente_empresa(
  id VARCHAR(4),
  cnpj VARCHAR(18),
  PRIMARY KEY(id),
  FOREIGN KEY(id) REFERENCES cliente(id)
);
CREATE TABLE taxi (
  placa VARCHAR(7),
  marca VARCHAR(30),
  modelo VARCHAR(30),
  anofab INTEGER,
  PRIMARY KEY(placa)
);
CREATE TABLE corrida (
  cliid VARCHAR(4),
  placa VARCHAR(7),
  dataPedido DATE,
  PRIMARY KEY(cliid, placa, dataPedido),
  FOREIGN KEY(cliid) REFERENCES cliente(id),
  FOREIGN KEY(placa) REFERENCES taxi(placa)
);
```

Com os seguintes dados

Cliente	
id	**nome**
1532	Asdrúbal
1755	Doriana
1780	Quincas
93	DinoTech
97	Proj

cliente_particular	
id	**cpf**
1532	448.754.253-44
1755	567.387.387-44
1780	576.456.123-55

corrida		
cliiid	**placa**	**datapedido**
1755	DAE6534	2003-02-15
97	JDM8776	2003-02-18

cliente_empresa	
id	**cnpj**
93	58.443.828/0001-02
97	44.876.234/7789-10

taxi			
placa	**marca**	**modelo**	**anofab**
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Pré-requisito

- Conhecimento em construção de consultas SQL:

```
SELECT <lista de colunas>
FROM <lista de tabelas>
[WHERE <condição>]
[GROUP BY <coluna_agrupar>]
[HAVING <condição_grupo>]
[ORDER BY <lista de atributos>]
```

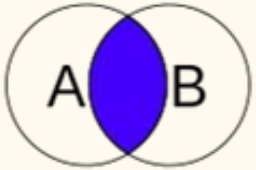
Joins

- Combina colunas de uma ou mais tabelas
- Existem diferentes tipos de junções em SQL
- Veremos aqui:
  - INNER
  - NATURAL
  - OUTER (RIGHT, LEFT OR FULL)

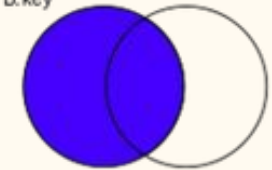
- Alguns SGBDs podem suportar outros tipos de junções não apresentadas aqui

SQL JOINS

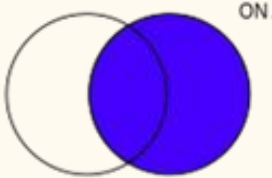
SELECT <fields>
FROM TableA A
INNER JOIN TableB B
ON A.key = B.key



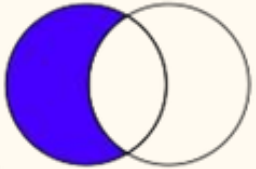
SELECT <fields>
FROM TableA A
LEFT JOIN TableB B
ON A.key = B.key



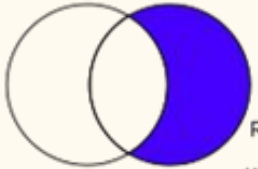
SELECT <fields>
FROM TableA A
RIGHT JOIN TableB B
ON A.key = B.key



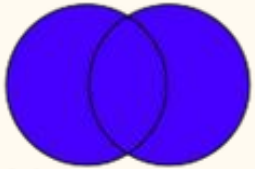
SELECT <fields>
FROM TableA A
LEFT JOIN TableB B
ON A.key = B.key
WHERE B.key IS NULL



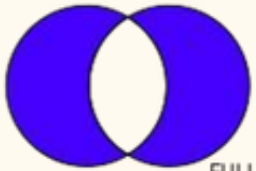
SELECT <fields>
FROM TableA A
RIGHT JOIN TableB B
ON A.key = B.key
WHERE A.key IS NULL



SELECT <fields>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.key = B.key

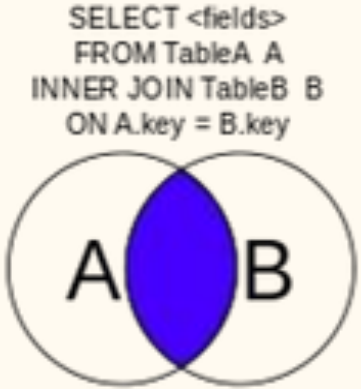


SELECT <fields>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.key = B.key
WHERE A.key IS NULL
OR B.key IS NULL



This work is licensed under a Creative Commons Attribution 3.0 Unported License.
Author: http://commons.wikimedia.org/wiki/User:Arbeck

INNER JOIN



- Ou apenas JOIN: seleciona as tuplas de acordo com a condição em ON.

- Exemplo: recuperar clientes que fizeram corridas de taxi

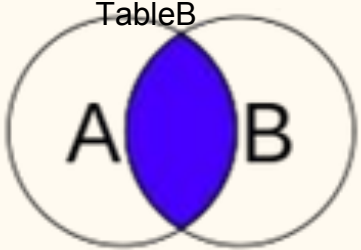
```
SELECT *
FROM cliente INNER JOIN corrida ON cliid=id;
```

- Equivalente a

```
SELECT *
FROM cliente, corrida
WHERE cliid=id;
```

id	nome	cliid	placa	datapedido
1755	Doriana	1755	DAE6534	2003-02-15
97	Proj	97	JDM8776	2003-02-18

SELECT <fields>
FROM TableA
NATURAL JOIN
TableB



NATURAL JOIN

- Seleciona as tuplas que tem valor equivalente para as colunas/atributos de mesmo nome (mesmo que na álgebra)
- Exemplo: recupera todos os taxis que fizeram corridas

```
SELECT *
FROM taxi NATURAL JOIN corrida;
```

a junção acontece em **placa**

placa	marca	modelo	anofab	cliid	datapedido
DAE6534	Ford	Fiesta	1999	1755	2003-02-15
JDM8776	Volkswagen	Santana	2002	97	2003-02-18

- Similar a:

```
SELECT *
FROM taxi as t INNER JOIN corrida as c ON
c.placa=t.placa;
```

- Neste caso, placa aparece duas vezes no esquema resultante (c.placa e t.placa)

LEFT OUTER JOIN

- Ou apenas LEFT JOIN: seleciona todas as tuplas da tabela à esquerda + tuplas da direita, desde que satisfaça a condição em ON.
- Isso permite que o lado direito tenha dados de valor NULL
- Exemplo: recupera todos os clientes e possíveis corridas que ele tenha feito

```
SELECT *
FROM cliente LEFT JOIN corrida ON cliid=id;
```

id	nome	cliid	placa	datapedido
1755	Doriana	1755	DAE6534	2003-02-15
97	Proj	97	JDM8776	2003-02-18
1532	Asdrúbal			
93	DinoTech			
1780	Quincas			

-

LEFT OUTER JOIN

- Podemos adicionar cláusula WHERE para recuperar apenas dados da esquerda que não têm correspondência com a direita.
- Isto é, apenas os dados em que o lado direito é NULL
- Exemplo: recupera todos os clientes que não fizeram corridas

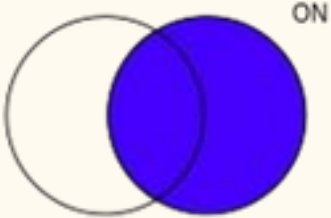
```
SELECT *
FROM cliente LEFT JOIN corrida ON cliid=id
WHERE cliid IS NULL;
```

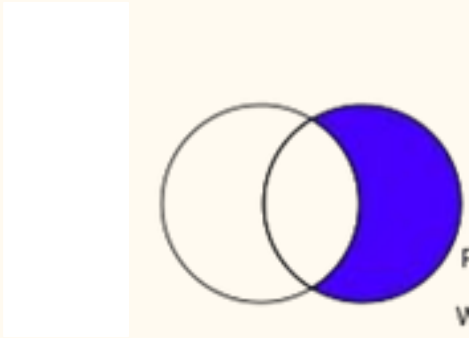
id	nome	cliid	placa	datapedido
1532	Asdrúbal			
93	DinoTech			
1780	Quincas			

RIGHT OUTER JOIN

- Simétrico ao LEFT OUTER JOIN

```
SELECT <fields>
FROM TableA A
RIGHT JOIN TableB B
ON A.key = B.key
```





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

FULL OUTER JOIN

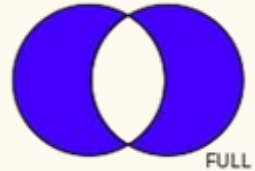
- Seleciona todas as tuplas que satisfazem a condição em ON + todas as tuplas das esquerda que não tiveram correspondências + todas as tuplas da direita que não tiveram correspondência
- Exemplo: recupera todos os clientes particulares, todas as corridas e associa corridas a clientes

id	cpf	cliid	placa	datapedido
1755	567.387.387-44	1755	DAE6534	2003-02-15
		97	JDM8776	2003-02-18
1532	448.754.253-44			
1780	576.456.123-55			

```
SELECT *
FROM cliente_particular FULL OUTER JOIN corrida ON cliid=id;
```

- Podemos notar:
  - o primeiro cliente_particular fez corrida;
  - os demais (ids 1532 e 1780) nunca fizeram corrida;
  - e ainda, existe uma corrida feita por algum cliente (cliid 97) que não é particular (neste contexto, é uma empresa).

FULL OUTER JOIN



- Similar ao LEFT e RIGHT JOIN podemos selecionar apenas as tuplas que não tem correspondência com o outro lado.
- Desta forma, permitiremos recuperar as tuplas que tem valores nulos ou na esquerda ou na direita.
- Exemplo: recupera corridas **não** feitas por clientes particulares e clientes particulares que **não** fizeram corridas

id	cpf	cliid	placa	datapedido
		97	JDM8776	2003-02-18
1532	448.754.253-44			
1780	576.456.123-55			

```
SELECT *
FROM cliente_particular FULL OUTER JOIN corrida ON cliid=id
WHERE id IS NULL OR cliid IS NULL;
```

Views

- é uma tabela virtual baseada em um conjunto de resultados de uma consulta SQL
- é uma tabela derivada de outras tabelas
- podemos usar visões para especificar tabelas que iremos usar com frequência
- Sintaxe:

```
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition
(...);
```

Exemplo

- Para corridas realizadas, os nomes e id dos clientes, dados do carro e da corrida.

```
CREATE VIEW cliTaxi AS
SELECT *
FROM taxi NATURAL JOIN corrida JOIN cliente ON cliid=id;
```

- Como recuperar os dados dessa visão?

```
SELECT * FROM cliTaxi;
```

placa	marca	modelo	anofab	cliid	datapedido	id	nome
DAE6534	Ford	Fiesta	1999	1755	2003-02-15	1755	Doriana
JDM8776	Wolkswagen	Santana	2002	97	2003-02-18	97	Proj

Exemplo - Possíveis consultas

- Nomes dos clientes que fizeram corridas:

  SELECT nome FROM clitaxi;

- modelos de carros que fizeram corridas:
  SELECT modelo FROM clitaxi;

- Data de corrida mais recente
  SELECT max(datapedido) FROM clitaxi;

- CNPj de clientes que fizeram corrida:
  SELECT cnpj FROM clitaxi AS t INNER JOIN cliente_empresa as c ON c.id=t.cliid;