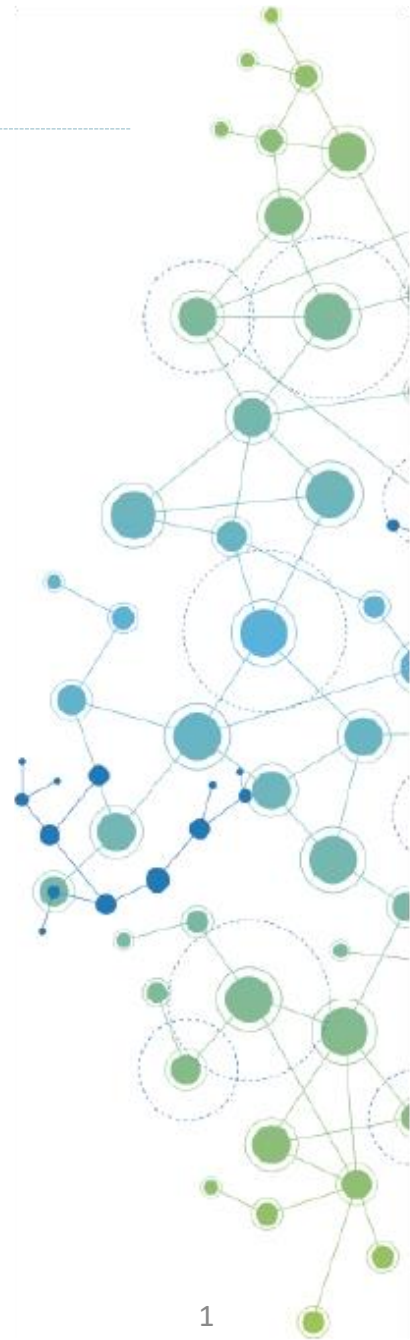


Utilizando Múltiplas Tabelas

JOIN



Extração de Dados

Utilização de múltiplas tabelas: JOINS



Na construção de um Banco de Dados são levados em consideração diversos fatores para otimização do desempenho na realização das consultas e do espaço de armazenamento das informações.

Por esses motivos, as informações são armazenadas em diferentes tabelas, mas com um campo de ligação entre elas.

id	data	id_produto
1	10/05/2019	11
2	11/05/2019	11
3	10/05/2019	31
4	10/05/2019	49

Tabela com as compras realizadas

- Muitos registros
- Campos menos extensos
- Atualização mais custosa

id_produto	nm_produto	cat_produto
11	Suco de laranja pasteurizado	Bebidas
12	Suco de uva pasteurizado	Bebidas
31	Fralda Infantil tamanho M	Higiene
49	Extrato de tomate	Molhos

Tabela de cadastro dos produtos

- Poucos registros
- Campos mais extensos
- Atualização menos custosa

Extração de Dados

Utilização de múltiplas tabelas: JOINS



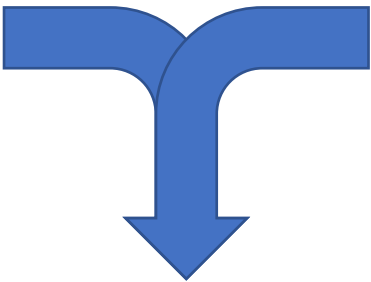
Como nosso foco é a análise de dados, assumiremos que os bancos de dados já estão devidamente construídos e com os dados disponíveis. Por isso, investiremos em aprender como utilizar as informações distribuídas em diversas tabelas diferentes.

tbcompras

id	data	id_produto
1	10/05/2019	11
2	11/05/2019	11
3	10/05/2019	31
4	10/05/2019	49

tbprodutos

id_produto	nm_produto	cat_produto
11	Suco de laranja pasteurizado	Bebidas
12	Suco de uva pasteurizado	Bebidas
31	Fralda Infantil tamanho M	Higiene
49	Extrato de tomate	Molhos



id	data	id_produto	id_produto	nm_produto	cat_produto
1	10/05/2019	11	11	Suco de laranja pasteurizado	Bebidas
2	11/05/2019	11	11	Suco de laranja pasteurizado	Bebidas
3	10/05/2019	31	31	Fralda Infantil tamanho M	Higiene
4	10/05/2019	49	49	Extrato de tomate	Molhos

Extração de Dados

Utilização de múltiplas tabelas: JOINS



No SQL a forma de unificarmos as informações de diferentes tabelas em um único resultado é utilizando os comandos **JOIN**.

No exemplo, temos a seguinte query:

```
SELECT
    tbcompras.*,
    tbprodutos.*
FROM
    tbcompras
JOIN tbprodutos
    ON tbcompras.id_produto = tbprodutos.id_produto
```

A palavra chave "ON" indica qual é o campo de ligação entre as duas tabelas.

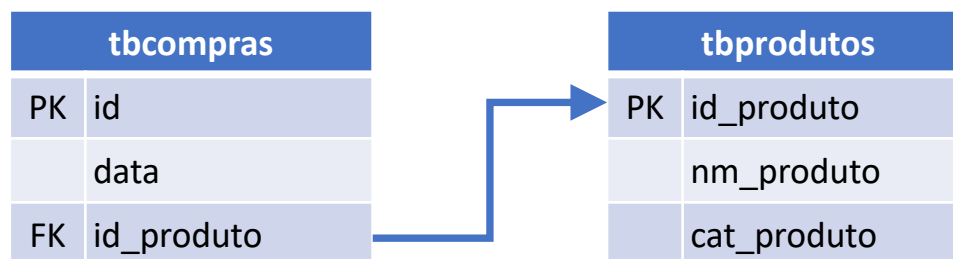
id	data	id_produto	id_produto	nm_produto	cat_produto
1	10/05/2019	11	11	Suco de laranja pasteurizado	Bebidas
2	11/05/2019	11	11	Suco de laranja pasteurizado	Bebidas
3	10/05/2019	31	31	Fralda Infantil tamanho M	Higiene
4	10/05/2019	49	49	Extrato de tomate	Molhos

Extração de Dados

Utilização de múltiplas tabelas: JOINS



O relacionamento entre as diversas tabelas pode ser apresentado em um **Entity Relationship Diagram** (ERD) , ou **Diagrama de Relacionamento das Entidades**. Abaixo temos o **ERD** das duas tabelas do exemplo anterior.



PK: Primary Key

- Identificador único de registros
- Existe em todas as tabelas

FK: Foreign Key

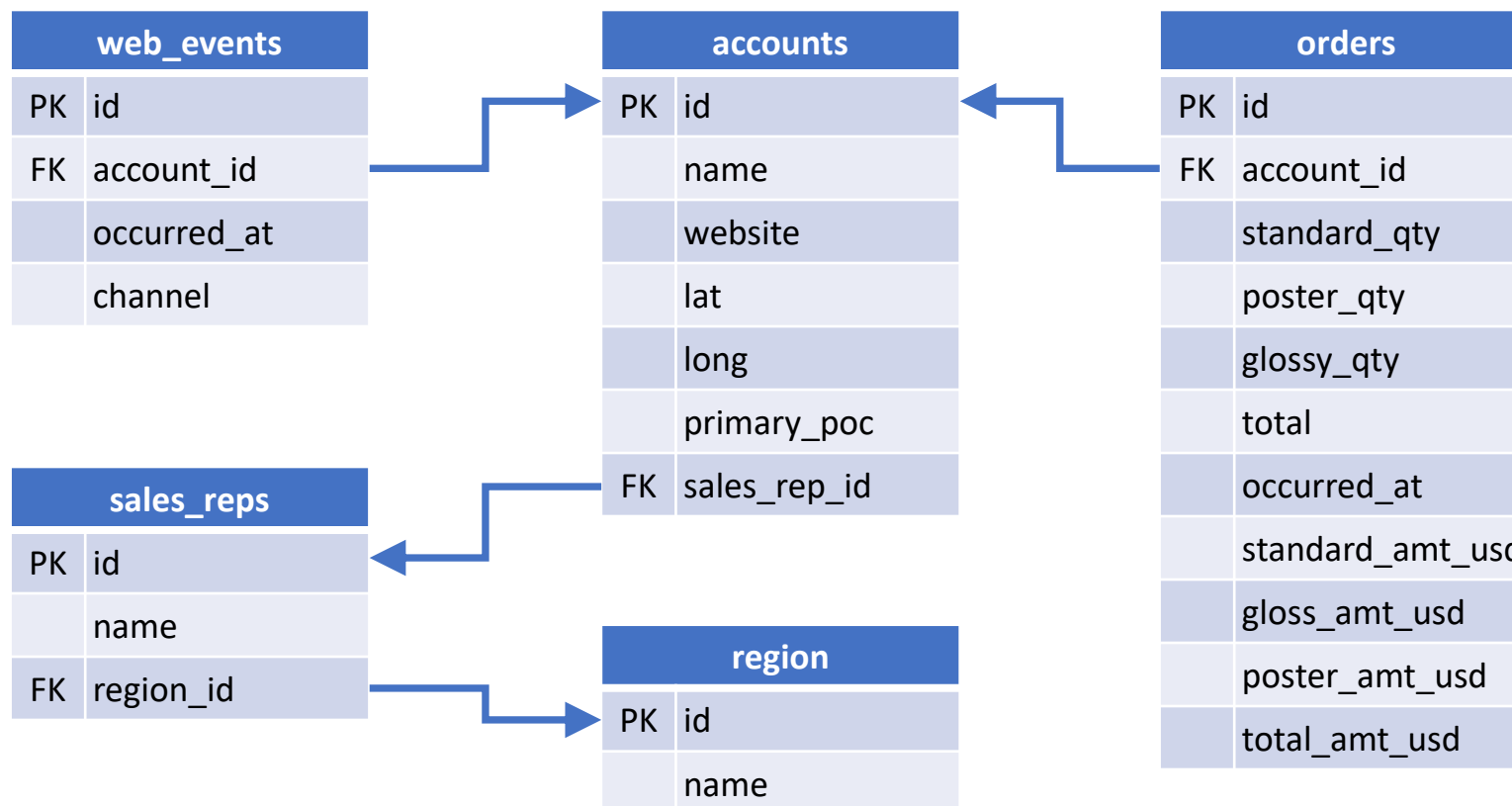
- Campo relacionado com um PK de outra tabela

Extração de Dados

Utilização de múltiplas tabelas: JOINS



Nos exemplos seguintes, utilizaremos o **Schema db_parchnposey** que contém um conjunto de 5 tabelas com o seguinte **Diagrama de Relacionamento das Entidades**:



PK: Primary Key

- Identificador único de registros
- Existe em todas as tabelas

FK: Foreign Key

- Campo relacionado com um PK de outra tabela

Extração de Dados

Utilização de múltiplas tabelas: JOINS



Essas 5 tabelas são de uma empresa fabricante de papel chamada **Parch and Posey**. Eles vendem **3 tipos de papéis**: normal, poster e glossy para grandes empresas do Fortune 100 e possui **50 representantes de vendas** (sales reps) em todo território Norte Americano, dividido em **4 regiões**. Suas campanhas de marketing incluem **Google, Facebook e Twitter**.

web_events			accounts			orders		
PK	id	Identificação única	PK	id	Identificação única	PK	id	Identificação do registro
FK	account_id	Id do cliente		name	Nome do cliente	FK	account_id	Identificação do cliente
	occurred_at	Momento do evento		website	Site do cliente		standard_qty	Qtde papel normal
	channel	Canal do evento		lat	Latitude da sede		poster_qty	Qtde papel poster
				long	Longitude da sede		glossy_qty	Qtde papel glossy
				primary_poc	Contato		total	Quantidade total
			FK	sales_rep_id	Id do representante de vendas		occurred_at	Data compra
sales_reps			region				standard_amt_usd	Valor papel normal
PK	id	Identificação única	PK	id	Identificação única		gloss_amt_usd	Valor papel glossy
	name	Nome representante		name	Nome da região		poster_amt_usd	Valor papel poster
FK	region_id	Id da região					total_amt_usd	Valor total

Extração de Dados

Utilização de múltiplas tabelas: JOINS



Para exercitar, vamos realizar uma query que tenha como resultado uma tabela com os campos:

- web_events.id
- web_events.occurred_at
- web_events.channel
- accounts.name
- accounts.website

```
SELECT TOP 100
```

```
    db_parchnposey.web_events.id,  
    db_parchnposey.web_events.occurred_at,  
    db_parchnposey.web_events.channel,  
    db_parchnposey.accounts.name,  
    db_parchnposey.accounts.website
```

```
FROM
```


```
    db_parchnposey.web_events
```

```
  JOIN db_parchnposey.accounts
```

```
    ON db_parchnposey.web_events.account_id = db_parchnposey.accounts.id
```

web_events

accounts

A diagram showing two tables, 'web_events' and 'accounts', with blue brackets above them. A large blue arrow points upwards from the SQL query towards the resulting table.

	id	occurred_at	channel	name	website
	SMALLINT	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)
1	1	2015-10-06T17:13:58.000Z	direct	Walmart	www.walmart.com
2	2	2015-11-05T03:08:26.000Z	direct	Walmart	www.walmart.com
3	3	2015-12-04T03:57:24.000Z	direct	Walmart	www.walmart.com
4	4	2016-01-02T00:55:03.000Z	direct	Walmart	www.walmart.com
5	5	2016-02-01T19:02:33.000Z	direct	Walmart	www.walmart.com
6	6	2016-03-02T15:15:22.000Z	direct	Walmart	www.walmart.com
7	7	2016-04-01T10:58:55.000Z	direct	Walmart	www.walmart.com
8	8	2016-05-01T15:26:44.000Z	direct	Walmart	www.walmart.com
9	9	2016-05-31T20:53:47.000Z	direct	Walmart	www.walmart.com
10	10	2016-06-30T12:09:45.000Z	direct	Walmart	www.walmart.com
11	11	2016-07-30T03:06:26.000Z	direct	Walmart	www.walmart.com
12	12	2016-08-28T06:42:42.000Z	direct	Walmart	www.walmart.com
13	13	2016-09-26T23:14:59.000Z	direct	Walmart	www.walmart.com



Extração de Dados

Utilização de múltiplas tabelas: JOINS



Predictiva.ai

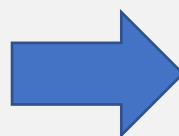
Vamos agora incluir os campos nome do representante de vendas (sales_reps.name) e a região (region.name) no resultado:

```
SELECT TOP 100
```

```
db_parchnposey.web_events.id,  
db_parchnposey.web_events.occurred_at,  
db_parchnposey.web_events.channel,  
db_parchnposey.accounts.name,  
db_parchnposey.accounts.website,  
db_parchnposey.sales_reps.name,  
db_parchnposey.region.name
```

```
FROM
```

```
db_parchnposey.web_events  
JOIN db_parchnposey.accounts  
    ON db_parchnposey.web_events.account_id = db_parchnposey.accounts.id  
JOIN db_parchnposey.sales_reps  
    ON db_parchnposey.accounts.sales_rep_id = db_parchnposey.sales_reps.id  
JOIN db_parchnposey.region  
    ON db_parchnposey.sales_reps.region_id = db_parchnposey.region.id
```



web_events				accounts		sales_rep region	
	id	occurred_at	channel	name	website	name	name
	SMALLINT	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)
1	1	2015-10-06T17:13:58.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
2	2	2015-11-05T03:08:26.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
3	3	2015-12-04T03:57:24.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
4	4	2016-01-02T00:55:03.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
5	5	2016-02-01T19:02:33.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
6	6	2016-03-02T15:15:22.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
7	7	2016-04-01T10:58:55.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
8	8	2016-05-01T15:26:44.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
9	9	2016-05-31T20:53:47.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
10	10	2016-06-30T12:09:45.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
11	11	2016-07-30T03:06:26.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
12	12	2016-08-28T06:42:42.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
13	13	2016-09-26T23:14:59.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
14	14	2016-10-26T20:21:09.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast

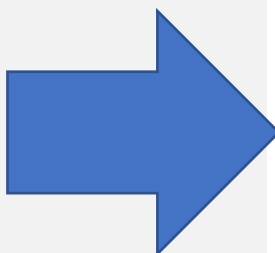
Extração de Dados

Utilização de múltiplas tabelas: JOINS



Uma forma de tornar o código mais limpo e mais fácil de ler é utilizar **ALIAS**, que são nomes mais curtos para cada tabela. Porém, perceba que a tabela resultado tem 3 campos com o nome "name".

```
SELECT TOP 100
  w.id,
  w.occurred_at,
  w.channel,
  a.name,
  a.website,
  s.name,
  r.name
FROM
  db_parchnposey.web_events as w
JOIN db_parchnposey.accounts as a
  ON w.account_id=a.id
JOIN db_parchnposey.sales_reps as s
  ON a.sales_rep_id=s.id
JOIN db_parchnposey.region as r
  ON s.region_id=r.id
```



	web_events			accounts		sales_rep	region
	id	occurred_at	channel	name	website	name	name
	SMALLINT	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)
1	1	2015-10-06T17:13:58.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
2	2	2015-11-05T03:08:26.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
3	3	2015-12-04T03:57:24.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
4	4	2016-01-02T00:55:03.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
5	5	2016-02-01T19:02:33.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
6	6	2016-03-02T15:15:22.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
7	7	2016-04-01T10:58:55.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
8	8	2016-05-01T15:26:44.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
9	9	2016-05-31T20:53:47.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
10	10	2016-06-30T12:09:45.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
11	11	2016-07-30T03:06:26.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
12	12	2016-08-28T06:42:42.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
13	13	2016-09-26T23:14:59.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
14	14	2016-10-26T20:21:00.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast

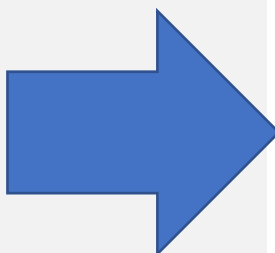
Extração de Dados

Utilização de múltiplas tabelas: JOINS



Vamos utilizar o **ALIAS** também para renomear os campos, como vimos anteriormente. Vamos renomear alguns campos da tabela de resultados:

```
SELECT TOP 100
  w.id,
  w.occurred_at,
  w.channel,
  a.name as account_name,
  a.website,
  s.name as sales_rep_name,
  r.name as region_name
FROM
  db_parchnposey.web_events as w
JOIN db_parchnposey.accounts as a
  ON w.account_id=a.id
JOIN db_parchnposey.sales_reps as s
  ON a.sales_rep_id=s.id
JOIN db_parchnposey.region as r
  ON s.region_id=r.id
```



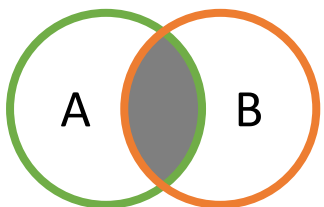
	web_events			accounts		sales_rep	region
	id	occurred_at	channel	name	website	name	name
	SMALLINT	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)	NVARCHAR (50)
1	1	2015-10-06T17:13:58.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
2	2	2015-11-05T03:08:26.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
3	3	2015-12-04T03:57:24.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
4	4	2016-01-02T00:55:03.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
5	5	2016-02-01T19:02:33.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
6	6	2016-03-02T15:15:22.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
7	7	2016-04-01T10:58:55.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
8	8	2016-05-01T15:26:44.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
9	9	2016-05-31T20:53:47.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
10	10	2016-06-30T12:09:45.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
11	11	2016-07-30T03:06:26.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
12	12	2016-08-28T06:42:42.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
13	13	2016-09-26T23:14:59.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast
14	14	2016-10-26T20:21:00.000Z	direct	Walmart	www.walmart.com	Samuel Racine	Northeast

Extração de Dados

Utilização de múltiplas tabelas: JOINS



O **JOIN** que utilizamos até agora é apenas 1 dos 4 tipos de JOINS disponíveis, o chamado **INNER JOIN**. Utilizando ele, são retornados apenas os registros que apresentam o campo de ligação igual nas duas tabelas.



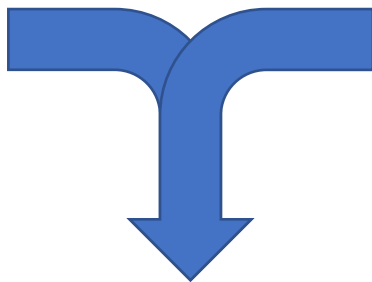
```
SELECT
  c.id,
  c.id_produto,
  p.id_produto,
  p.cat_produto
FROM
  tbcompras as c
  INNER JOIN tbprodutos as p
    ON c.id_produto = p.id_produto
```

Tabela A

id	id_produto
1	11
3	31
4	49
5	55

Tabela B

id_produto	cat_produto
11	Bebidas
12	Bebidas
31	Higiene
49	Molhos



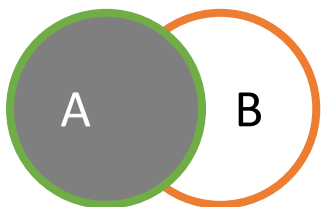
id	id_produto	id_produto	cat_produto
1	11	11	Bebidas
3	31	31	Higiene
4	49	49	Molhos

Extração de Dados

Utilização de múltiplas tabelas: JOINS



O **LEFT JOIN** retorna todos os registros da tabela A e apenas os registros da tabela B que possuam o campo de ligação com o mesmo valor daqueles na tabela A.



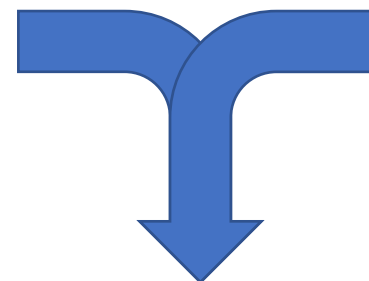
```
SELECT
  c.id,
  c.id_produto,
  p.id_produto,
  p.cat_produto
FROM
  tbcompras as c
  LEFT JOIN tbprodutos as p
    ON c.id_produto = p.id_produto
```

Tabela A

id	id_produto
1	11
3	31
4	49
5	55

Tabela B

id_produto	cat_produto
11	Bebidas
12	Bebidas
31	Higiene
49	Molhos



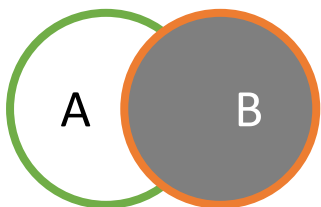
id	id_produto	id_produto	cat_produto
1	11	11	Bebidas
3	31	31	Higiene
4	49	49	Molhos
5	55	NULL	NULL

Extração de Dados

Utilização de múltiplas tabelas: JOINS



O **RIGHT JOIN** faz exatamente o inverso do LEFT JOIN, ou seja, retorna todos os registros da tabela B e apenas os registros da tabela A que possuam o campo de ligação com o mesmo valor daqueles na tabela B.



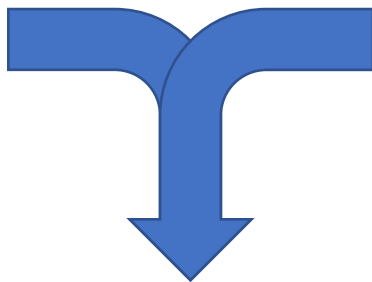
```
SELECT
  c.id,
  c.id_produto,
  p.id_produto,
  p.cat_produto
FROM
  tbcompras as c
  RIGHT JOIN tbprodutos as p
  ON c.id_produto = p.id_produto
```

Tabela A

id	id_produto
1	11
3	31
4	49
5	55

Tabela B

id_produto	cat_produto
11	Bebidas
12	Bebidas
31	Higiene
49	Molhos



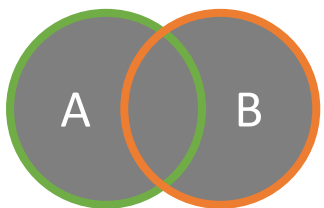
id	id_produto	id_produto	cat_produto
1	11	11	Bebidas
NULL	NULL	12	Bebidas
3	31	31	Higiene
4	49	49	Molhos

Extração de Dados

Utilização de múltiplas tabelas: JOINS



O **FULL OUTER JOIN** retorna todos os registros das tabelas A e B, e aqueles que não possuírem o campo de ligação com o mesmo valor são automaticamente preenchidos com **NULL**.



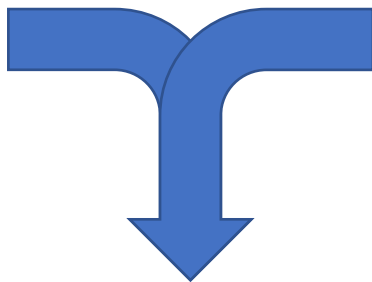
```
SELECT
  c.id,
  c.id_produto,
  p.id_produto,
  p.cat_produto
FROM
  tbcompras as c
  FULL OUTER JOIN tbprodutos as p
    ON c.id_produto = p.id_produto
```

Tabela A

id	id_produto
1	11
3	31
4	49
5	55

Tabela B

id_produto	cat_produto
11	Bebidas
12	Bebidas
31	Higiene
49	Molhos



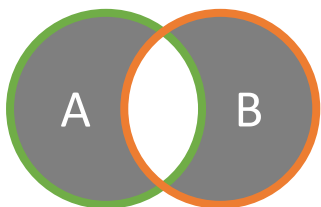
id	id_produto	id_produto	cat_produto
1	11	11	Bebidas
NULL	NULL	12	Bebidas
3	31	31	Higiene
4	49	49	Molhos
5	55	NULL	NULL

Extração de Dados

Utilização de múltiplas tabelas: JOINS



Além de unificar campos de diferentes tabelas, o **JOIN** pode também ser utilizado para localizar registros que não tenham conexão entre as duas tabelas. Essa utilização é bastante útil para identificar potenciais problemas no conteúdo dos campos como missing e valores não padronizados, por exemplo.



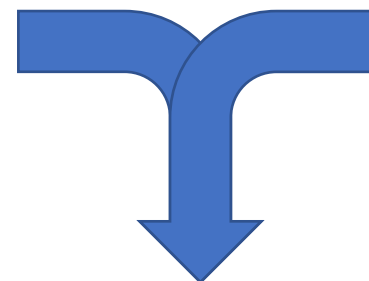
```
SELECT
  c.id,
  c.id_produto,
  p.id_produto,
  p.cat_produto
FROM
  tbcompras as c
  FULL OUTER JOIN tbprodutos as p
    ON c.id_produto = p.id_produto
WHERE
  c.id_produto IS NULL OR p.id_produto IS NULL
```

Tabela A

id	id_produto
1	11
3	31
4	49
5	55

Tabela B

id_produto	cat_produto
11	Bebidas
12	Bebidas
31	Higiene
49	Molhos



id	id_produto	id_produto	cat_produto
NULL	NULL	12	Bebidas
5	55	NULL	NULL

Extração de Dados

Utilização de múltiplas tabelas: JOINS



Os **JOINS** também podem ser utilizados com múltiplas comparações na cláusula **ON**, e inclusive com uma comparação diferente da igualdade.

Na prática o resultado é semelhante ao de um filtro, e como na ordem de execução do SQL as cláusulas **JOIN** são realizadas primeiro, utilizar essa funcionalidade pode trazer ganhos de performance.

```
SELECT
  o.id,
  o.occurred_at as order_date,
  w.*
FROM db_parchnposey.orders o
  LEFT JOIN db_parchnposey.web_events w
    ON w.account_id = o.account_id
   AND w.occurred_at < o.occurred_at
ORDER BY
  o.account_id , o.occurred_at,
  w.occurred_at
```



	123 id 🔍	🕒 order_date 🔍	123 id 🔍	123 account_id 🔍	🕒 occurred_at 🔍	ABC channel 🔍
1	4,308.00	2015-12-04 02:01:09	4,394.00	1,001.00	2015-10-06 01:22:11	facebook
2	4,308.00	2015-12-04 02:01:09	1.00	1,001.00	2015-10-06 14:13:58	direct
3	4,308.00	2015-12-04 02:01:09	4,395.00	1,001.00	2015-10-22 03:02:47	organic
4	4,308.00	2015-12-04 02:01:09	4,396.00	1,001.00	2015-10-22 12:04:20	adwords
5	4,308.00	2015-12-04 02:01:09	2.00	1,001.00	2015-11-05 01:08:26	direct
6	4,308.00	2015-12-04 02:01:09	4,397.00	1,001.00	2015-11-05 15:18:54	direct
7	4,308.00	2015-12-04 02:01:09	3.00	1,001.00	2015-12-04 01:57:24	direct
8	3.00	2015-12-04 02:21:55	4,394.00	1,001.00	2015-10-06 01:22:11	facebook
9	3.00	2015-12-04 02:21:55	1.00	1,001.00	2015-10-06 14:13:58	direct
10	3.00	2015-12-04 02:21:55	4,395.00	1,001.00	2015-10-22 03:02:47	organic
11	3.00	2015-12-04 02:21:55	4,396.00	1,001.00	2015-10-22 12:04:20	adwords
12	3.00	2015-12-04 02:21:55	2.00	1,001.00	2015-11-05 01:08:26	direct

Extração de Dados

Utilização de múltiplas tabelas: **SELF JOINS**

Outra funcionalidade interessante no SQL é a do **SELF JOIN**, que significa fazer um **JOIN** de uma tabela com ela mesma.

Um uso bastante comum dessa funcionalidade ocorre quando queremos selecionar registros em um **espaço de tempo**, por exemplo. Dessa forma, podemos selecionar os registros com uma **data de referência** e uma **janela pré-definida**, como no exemplo abaixo:

```
SELECT o1.id AS o1_id,  
       o1.account_id AS o1_account_id,  
       o1.occurred_at AS o1_occurred_at,  
       o2.id AS o2_id,  
       o2.account_id AS o2_account_id,  
       o2.occurred_at AS o2_occurred_at  
FROM db_parchnposey.orders o1  
LEFT JOIN db_parchnposey.orders o2  
  ON o1.account_id = o2.account_id  
  AND o2.occurred_at > o1.occurred_at  
  AND o2.occurred_at <=  
    dateadd(dd, 30, o1.occurred_at)  
ORDER BY o1.account_id, o1.occurred_at
```



	o1_id	o1_account_id	o1_occurred_at	o2_id	o2_account_id	o2_occurred_at
1	1.00	1,001.00	2015-10-06 14:31:14	2.00	1,001.00	2015-11-05 01:34:33
2	1.00	1,001.00	2015-10-06 14:31:14	4,307.00	1,001.00	2015-11-05 01:25:21
3	4,307.00	1,001.00	2015-11-05 01:25:21	2.00	1,001.00	2015-11-05 01:34:33
4	4,307.00	1,001.00	2015-11-05 01:25:21	3.00	1,001.00	2015-12-04 02:21:55
5	4,307.00	1,001.00	2015-11-05 01:25:21	4,308.00	1,001.00	2015-12-04 02:01:09
6	2.00	1,001.00	2015-11-05 01:34:33	3.00	1,001.00	2015-12-04 02:21:55
7	2.00	1,001.00	2015-11-05 01:34:33	4,308.00	1,001.00	2015-12-04 02:01:09
8	4,308.00	1,001.00	2015-12-04 02:01:09	3.00	1,001.00	2015-12-04 02:21:55
9	4,308.00	1,001.00	2015-12-04 02:01:09	4.00	1,001.00	2016-01-01 23:18:24
10	4,308.00	1,001.00	2015-12-04 02:01:09	4,309.00	1,001.00	2016-01-01 22:59:09
11	3.00	1,001.00	2015-12-04 02:21:55	4.00	1,001.00	2016-01-01 23:18:24
12	3.00	1,001.00	2015-12-04 02:21:55	4,309.00	1,001.00	2016-01-01 22:59:09

Extração de Dados

Utilização de múltiplas tabelas: UNION



Quando utilizamos os **JOINS** estamos incluindo campos de diversas tabelas em uma tabela final, mas algumas vezes precisamos "empilhar" os registros de diversas queries.

Nesse caso, devemos utilizar o **UNION**:

```
SELECT
    a.id,
    a.name,
    ('accounts') as tipo
FROM
    db_parchnposey.accounts a
UNION
SELECT
    s.id,
    s.name,
    ('sales reps') as tipo
FROM
    db_parchnposey.sales_reps s
```



	123 id	ABC name	ABC tipo
343	4,421.00	Eversource Energy	accounts
344	4,431.00	Franklin Resources	accounts
345	4,441.00	Masco	accounts
346	4,451.00	Lithia Motors	accounts
347	4,461.00	KKR	accounts
348	4,471.00	Oneok	accounts
349	4,481.00	Newmont Mining	accounts
350	4,491.00	PPL	accounts
351	4,501.00	SpartanNash	accounts
352	321,500.00	Samuel Racine	sales reps
353	321,510.00	Eugena Esser	sales reps
354	321,520.00	Michel Averette	sales reps
355	321,530.00	Renetta Carew	sales reps
356	321,540.00	Cara Clarke	sales reps
357	321,550.00	Lavera Oles	sales reps
358	321,560.00	Elba Felder	sales reps
359	321,570.00	Shawanda Selke	sales reps
360	321,580.00	Sibyl Lauria	sales reps



Preditiva.ai