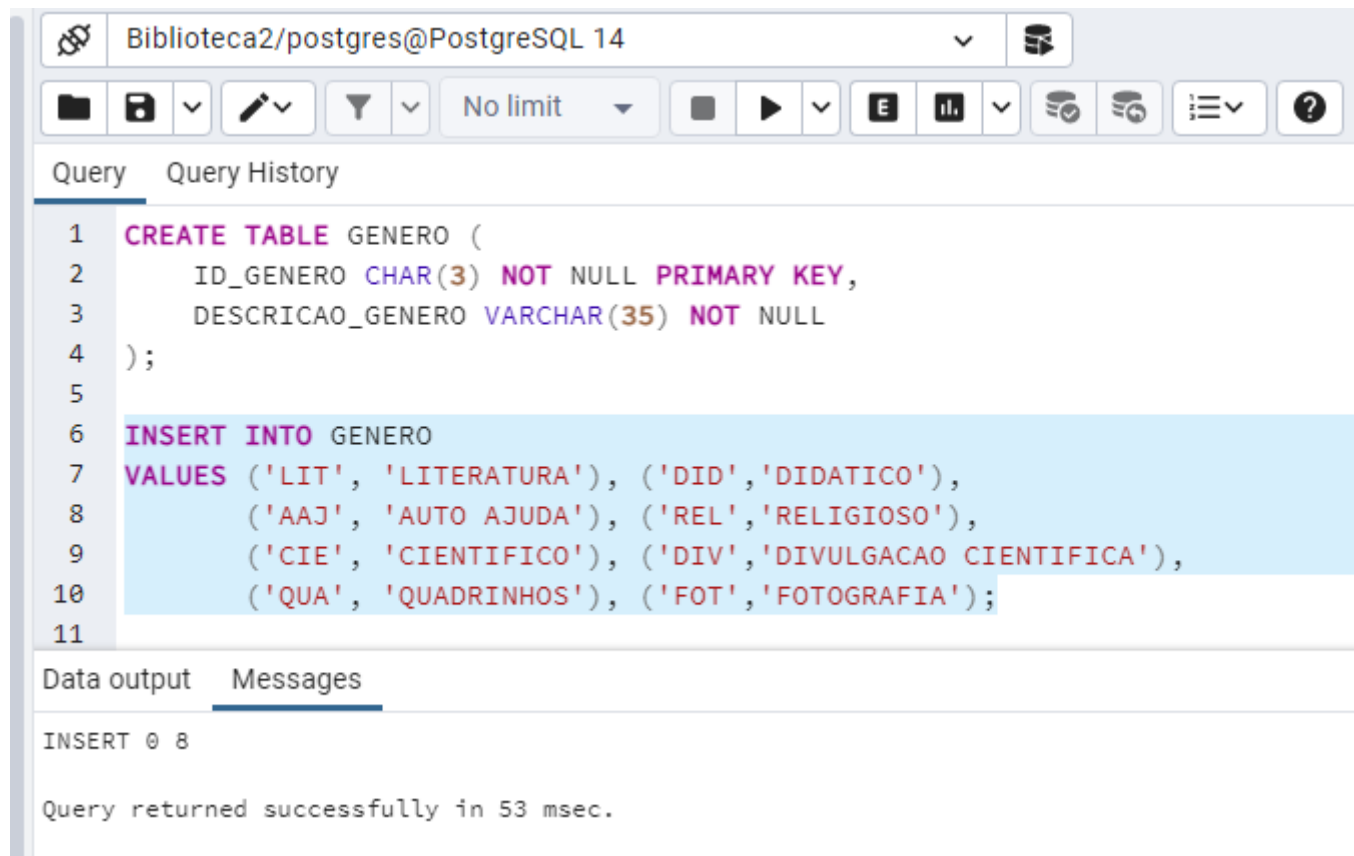


Banco de Dados 2

10 – Particionamento de Tabelas

Banco de Dados Biblioteca



The screenshot shows a PostgreSQL client window titled 'Biblioteca2/postgres@PostgreSQL 14'. The interface includes a toolbar with icons for file operations, query execution, and settings. The main area displays a SQL query with line numbers 1 through 11. The query consists of a `CREATE TABLE` statement for a table named `GENERO` and an `INSERT INTO` statement. The `INSERT` statement is highlighted with a light blue background. Below the query, the 'Messages' tab is active, showing the output 'INSERT 0 8' and a confirmation message: 'Query returned successfully in 53 msec.'

```
1 CREATE TABLE GENERO (  
2     ID_GENERO CHAR(3) NOT NULL PRIMARY KEY,  
3     DESCRICAO_GENERO VARCHAR(35) NOT NULL  
4 );  
5  
6 INSERT INTO GENERO  
7 VALUES ('LIT', 'LITERATURA'), ('DID', 'DIDATICO'),  
8         ('AAJ', 'AUTO AJUDA'), ('REL', 'RELIGIOSO'),  
9         ('CIE', 'CIENTIFICO'), ('DIV', 'DIVULGACAO CIENTIFICA'),  
10        ('QUA', 'QUADRINHOS'), ('FOT', 'FOTOGRAFIA');  
11
```

INSERT 0 8

Query returned successfully in 53 msec.

Banco de Dados Biblioteca

```
11
12 CREATE TABLE OBRA (
13     NR_OBRA SERIAL NOT NULL PRIMARY KEY,
14     TITULO VARCHAR(45) NOT NULL,
15     ID_GENERO CHAR(3) NOT NULL,
16
17     FOREIGN KEY(ID_GENERO) REFERENCES GENERO (ID_GENERO)
18     ON DELETE RESTRICT
19 );
20
21 INSERT INTO OBRA (TITULO, ID_GENERO) VALUES
22 ('POEMA DE MIO CID','LIT'), ('OS TRÊS MOSQUETEIROS','LIT'),
23 ('O CONDE DE MONTECRISTO','LIT'), ('HELENA','LIT'),
24 ('INTELIGENCIA ARTIFICIAL','CIE'), ('O PENSAMENTO POSITIVO DO FOD@-SE','AAJ'),
25 ('COMO FAZER AMIGOS E INFLUENCIAR PESSOAS','AAJ'),('TEORIA DO CAOS','CIE'),
26 ('ASTERIX E A FOICE DE OURO','QUA');
27
28
```

Data output Messages

INSERT 0 9

Query returned successfully in 58 msec.

Banco de Dados Biblioteca

```
27
28 CREATE TABLE AUTOR (
29     ID_AUTOR CHAR(5) NOT NULL PRIMARY KEY,
30     NOME_AUTOR VARCHAR (35) NOT NULL
31 );
32
33 INSERT INTO AUTOR VALUES
34 ('DESCO','***** DESCONHECIDO *****'), ('ADUMA','ALEXANDRE DUMAS'),
35 ('MASSI','MACHADO DE ASSIS'),('RUSSE','STUART RUSSEL'),
36 ('NORVI','PETER NORVIG'), ('JMORP','JOSEPH MORPHY'),
37 ('JGLEI','JAMES GLEICK'), ('GOSCI','RENE GOSCINY'),
38 ('UDERZ','ALBERT UDERZO');
39
```

Data output Messages

INSERT 0 9

Query returned successfully in 58 msec.

Banco de Dados Biblioteca

```
40 CREATE TABLE AUTORIA (  
41     NR_OBRA INTEGER NOT NULL,  
42     ID_AUTOR CHAR(5) NOT NULL,  
43  
44     PRIMARY KEY (NR_OBRA, ID_AUTOR),  
45  
46     FOREIGN KEY (NR_OBRA) REFERENCES OBRA (NR_OBRA)  
47     ON DELETE RESTRICT,  
48  
49     FOREIGN KEY (ID_AUTOR) REFERENCES AUTOR (ID_AUTOR)  
50     ON DELETE RESTRICT  
51 );  
52  
53 INSERT INTO AUTORIA VALUES (1, 'DESCO'), (2, 'ADUMA'), (3, 'ADUMA'),  
54 (4, 'MASSI'), (5, 'RUSSE'), (5, 'NORVI'), (6, 'JMORP'), (7, 'JMORP'),  
55 (8, 'JGLEI'), (9, 'GOSCI'), (9, 'UDERZ');  
56
```

Data output Messages

INSERT 0 11

Query returned successfully in 105 msec.

Banco de Dados Biblioteca

```
52  '
53  INSERT INTO AUTORIA VALUES (1,'DESCO'),(2,'ADUMA'),(3,'ADUMA'),
54  (4,'MASSI'),(5,'RUSSE'),(5,'NORVI'), (6,'JMORP'), (7,'JMORP'),
55  (8,'JGLEI'),(9,'GOSCI'),(9,'UDERZ');
56
57  CREATE TABLE EDITORA (
58      ID_EDITORA CHAR(3) NOT NULL PRIMARY KEY,
59      NOME_EDITORA VARCHAR(40) NOT NULL
60  );
61
62  INSERT INTO EDITORA VALUES ('EBA','EBAL - EDITORA BRASIL AMERICA'),
63  ('RGE','RIO GRAFICA EDITORA'),('BLO','BLOCH EDITORES'),
64  ('ROC','ROCCO EDITORA'),('SEX','SEXTANTE');
65
```

Data output Messages

INSERT 0 5

Query returned successfully in 60 msec.

Banco de Dados Biblioteca

```
65
66 CREATE TABLE LIVRO (
67     NR_LIVRO SERIAL NOT NULL PRIMARY KEY,
68     DT_AQUISICAO DATE NOT NULL,
69     ID_EDITORA CHAR(3) NOT NULL,
70
71     FOREIGN KEY (ID_EDITORA) REFERENCES EDITORA (ID_EDITORA)
72     ON DELETE RESTRICT
73 );
74
75 INSERT INTO LIVRO (DT_AQUISICAO, ID_EDITORA) VALUES
76 ('2025-01-02', 'EBA'), ('2025-02-21', 'EBA'), ('2025-01-30', 'BLO'),
77 ('2025-02-14', 'BLO'), ('2025-03-25', 'RGE'), ('2025-08-29', 'RGE'),
78 ('2026-04-03', 'BLO'), ('2026-09-22', 'ROC'), ('2026-09-23', 'ROC'),
79 ('2026-09-28', 'ROC'), ('2026-10-03', 'ROC'), ('2026-11-02', 'SEX');
80
```

Data output Messages

INSERT 0 12

Query returned successfully in 63 msec.

Banco de Dados Biblioteca

```
81 CREATE TABLE COMPOSICAO (  
82     NR_OBRA INTEGER NOT NULL,  
83     NR_LIVRO INTEGER NOT NULL,  
84  
85     PRIMARY KEY(NR_OBRA, NR_LIVRO),  
86  
87     FOREIGN KEY (NR_OBRA) REFERENCES OBRA (NR_OBRA)  
88     ON DELETE RESTRICT,  
89  
90     FOREIGN KEY (NR_LIVRO) REFERENCES LIVRO (NR_LIVRO)  
91     ON DELETE RESTRICT  
92 );  
93  
94 INSERT INTO COMPOSICAO VALUES (1,1), (1,2), (2,3), (3,3), (4,4), (5,5),  
95 (5,6), (5,7), (6,8), (7,8), (8,9), (8,10), (8,11), (9,12);  
96
```

Data output Messages

INSERT 0 14

Query returned successfully in 64 msec.

Banco de Dados Biblioteca

```
96
97 CREATE TABLE USUARIO (
98     MATRICULA SERIAL NOT NULL PRIMARY KEY,
99     NOME_USUARIO VARCHAR(45) NOT NULL,
100     SEXO CHAR(1) NOT NULL,
101     DT_NASCIMENTO DATE NOT NULL
102 );
103
104 INSERT INTO USUARIO (NOME_USUARIO, SEXO, DT_NASCIMENTO) VALUES
105 ('ASDRUBAL SOARES RIBEIRO', 'M', '1961-11-09'),
106 ('JAMBIRA TIMBIRAS', 'F', '1999-01-23'),
107 ('CECILIA GARFUNKEL', 'F', '2009-05-10');
108
```

Data output Messages

INSERT 0 3

Query returned successfully in 90 msec.

Banco de Dados Biblioteca

```
108
109 CREATE TABLE MOVIMENTACAO (
110     NR_LIVRO INTEGER NOT NULL,
111     MATRICULA INTEGER NOT NULL,
112     DT_EMPRESTIMO DATE NOT NULL,
113     DT_PREVISTA_DEVOLUCAO DATE,
114     DT_EFETIVA_DEVOLUCAO DATE,
115
116     PRIMARY KEY(NR_LIVRO, MATRICULA, DT_EMPRESTIMO),
117
118     FOREIGN KEY (NR_LIVRO) REFERENCES LIVRO (NR_LIVRO)
119     ON DELETE RESTRICT,
120
121     FOREIGN KEY (MATRICULA) REFERENCES USUARIO(MATRICULA)
122     ON DELETE RESTRICT
123 );
124
```

Particionamento de Tabelas



Particionamento de Tabelas

- O **particionamento de tabelas** em um **banco de dados PostgreSQL** é uma das responsabilidades dos DBAs (Database Administrator).
- Tem por objetivo **melhorar o gerenciamento de um banco de dados através da divisão de tabelas com grande volume de dados em partes menores**, e **mais fáceis de serem gerenciadas**.

Particionamento de Tabelas

- A ideia de **particionar tabela de dados** é útil para **dividirmos uma grande tabela em partições (ou tabelas) menores**, de forma a tornar **as consultas de geração de relatório e estatísticas menos onerosas** para o banco de dados.
- **Particionar uma tabela de dados** consiste em **dividir um grande volume de dados que seria armazenado em uma única tabela em partições (ou tabelas) menores**, onde cada uma delas atende a um **conjunto específico de dados definido a partir de critérios/regras bem definidas para divisão**.
- Assim, **ao invés de consultarmos um único “tabelão” com milhões de registros**, nossa consulta pode **acessar apenas a partição que contém os dados que desejamos**, gerando um **custo muito menor para o banco de dados** e conseqüentemente, **uma consulta mais rápida e eficiente**.

Particionamento de Tabelas

- Antes de darmos seguimento com as **partições**, precisamos entender primeiro o **processo de implementação de herança** que ocorre entre as **tabelas do PostgreSQL**, o que pode ser de grande utilidade ao criarmos nossas bases de dados.
- A **herança** é um **conceito de bancos de dados orientados a objeto**, que abre possibilidades interessantes para os **projetos de bancos de dados**, onde no **PostgreSQL** temos que uma tabela pode herdar de nenhuma ou de várias tabelas.

Particionamento de Tabelas

```
124  
125 -- CRIANDO UMA PARTIÇÃO DA TABELA MOVIMENTACAO (ATRAVÉS DE HERANÇA)  
126 CREATE TABLE MOVIMENTACAO_2025 (  
127     CHECK (DT_EMPRESTIMO >= '2025-01-01' AND DT_EMPRESTIMO <= '2025-12-31')  
128 ) INHERITS(MOVIMENTACAO);  
129
```

Data output Messages

CREATE TABLE

Query returned successfully in 90 msec.

- MOVIMENTACAO_2025 herda de MOVIMENTACAO.

Particionamento de Tabelas

```
124
125 -- CRIANDO UMA PARTIÇÃO DA TABELA MOVIMENTACAO (ATRAVÉS DE HERANÇA)
126 CREATE TABLE MOVIMENTACAO_2025 (
127     CHECK (DT_EMPRESTIMO >= '2025-01-01' AND DT_EMPRESTIMO <= '2025-12-31')
128 ) INHERITS(MOVIMENTACAO);
129
130 CREATE TABLE MOVIMENTACAO_2026 (
131     CHECK (DT_EMPRESTIMO >= '2026-01-01' AND DT_EMPRESTIMO <= '2026-12-31')
132 ) INHERITS(MOVIMENTACAO);
133
134
```

Data output Messages

CREATE TABLE

Query returned successfully in 61 msec.

Particionamento de Tabelas

```
124
125 -- CRIANDO UMA PARTIÇÃO DA TABELA MOVIMENTACAO (ATRAVÉS DE HERANÇA)
126 CREATE TABLE MOVIMENTACAO_2025 (
127     CHECK (DT_EMPRESTIMO >= '2025-01-01' AND DT_EMPRESTIMO <= '2025-12-31')
128 )INHERITS(MOVIMENTACAO);
129
130 CREATE TABLE MOVIMENTACAO_2026 (
131     CHECK (DT_EMPRESTIMO >= '2026-01-01' AND DT_EMPRESTIMO <= '2026-12-31')
132 )INHERITS(MOVIMENTACAO);
133
134 CREATE TABLE MOVIMENTACAO_2027 (
135     CHECK (DT_EMPRESTIMO >= '2027-01-01' AND DT_EMPRESTIMO <= '2027-12-31')
136 )INHERITS(MOVIMENTACAO);
137
```

Data output Messages

CREATE TABLE

Query returned successfully in 95 msec.

Particionamento de Tabelas

```
133
134 CREATE TABLE MOVIMENTACAO_2027 (
135     CHECK (DT_EMPRESTIMO >= '2027-01-01' AND DT_EMPRESTIMO <= '2027-12-31')
136 ) INHERITS(MOVIMENTACAO);
137
138 -----
139 -- Agora que temos 3 partições criadas para tabela MOVIMENTACAO devemos também --
140 -- adicionar índices para cada uma das tabelas filhas (MOVIMENTACAO_2025, --
141 -- MOVIMENTACAO_2026 e MOVIMENTACAO_2027. Dessa forma agilizaremos as consultas--
142 -- sobre o conteúdo destas tabelas filhas. Os índices serão criados a partir --
143 -- da coluna DT_EMPRESTIMO -----
144 -----
145
146 CREATE INDEX MOVIMENTACAO_2025_IDX ON MOVIMENTACAO_2025 (DT_EMPRESTIMO);
147
```

Data output Messages

CREATE INDEX

Query returned successfully in 110 msec.

Particionamento de Tabelas

```
145  
146 CREATE INDEX MOVIMENTACAO_2025_IDX ON MOVIMENTACAO_2025 (DT_EMPRESTIMO);  
147 CREATE INDEX MOVIMENTACAO_2026_IDX ON MOVIMENTACAO_2026 (DT_EMPRESTIMO);  
148 CREATE INDEX MOVIMENTACAO_2027_IDX ON MOVIMENTACAO_2027 (DT_EMPRESTIMO);  
149
```

Data output Messages

CREATE INDEX

Query returned successfully in 116 msec.

Particionamento de Tabelas

```
145
146 CREATE INDEX MOVIMENTACAO_2025_IDX ON MOVIMENTACAO_2025 (DT_EMPRESTIMO);
147 CREATE INDEX MOVIMENTACAO_2026_IDX ON MOVIMENTACAO_2026 (DT_EMPRESTIMO);
148 CREATE INDEX MOVIMENTACAO_2027_IDX ON MOVIMENTACAO_2027 (DT_EMPRESTIMO);
149
150 CREATE INDEX movimentacao_2025_matricula_idx ON movimentacao_2025 (matricula);
151 CREATE INDEX movimentacao_2026_matricula_idx ON movimentacao_2026 (matricula);
152 CREATE INDEX movimentacao_2027_matricula_idx ON movimentacao_2027 (matricula);
153
154
```

Data output Messages

CREATE INDEX

Query returned successfully in 115 msec.

Particionamento de Tabelas

```
145  
146 CREATE INDEX MOVIMENTACAO_2025_IDX ON MOVIMENTACAO_2025 (DT_EMPRESTIMO);  
147 CREATE INDEX MOVIMENTACAO_2026_IDX ON MOVIMENTACAO_2026 (DT_EMPRESTIMO);  
148 CREATE INDEX MOVIMENTACAO_2027_IDX ON MOVIMENTACAO_2027 (DT_EMPRESTIMO);  
149  
150 CREATE INDEX movimentacao_2025_matricula_idx ON movimentacao_2025 (matricula);  
151 CREATE INDEX movimentacao_2026_matricula_idx ON movimentacao_2026 (matricula);  
152 CREATE INDEX movimentacao_2027_matricula_idx ON movimentacao_2027 (matricula);  
153  
154 CREATE INDEX movimentacao_2025_livro_idx ON movimentacao_2025 (nr_livro);  
155 CREATE INDEX movimentacao_2026_livro_idx ON movimentacao_2026 (nr_livro);  
156 CREATE INDEX movimentacao_2027_livro_idx ON movimentacao_2027 (nr_livro);  
157
```

Data output Messages

CREATE INDEX

Query returned successfully in 63 msec.

Particionamento de Tabelas

- Estes índices para NR_LIVRO e MATRICULA somente devem ser criados quando existir a possibilidade de frequente consultas por essas colunas isoladamente.
- **PRIMARY KEY (nr_livro, matricula, dt_emprestimo)**
- Ou seja: já existe um índice único (B-Tree) que cobre essas três colunas, na ordem: **(nr_livro, matricula, dt_emprestimo)**.
- E como cada **tabela filha** (**MOVIMENTACAO_2025, MOVIMENTACAO_2026, etc.**) herda os campos, esse índice também precisa ser criado nelas quando você define a PK. Portanto, **índices em nr_livro e matricula já existem — mas compostos com dt_emprestimo.**

Particionamento de Tabelas

- Mesmo que já exista o índice da PK, **índices adicionais** podem ser úteis em alguns cenários:
- Consultas filtrando apenas por uma coluna: **SELECT * FROM movimentacao_2025 WHERE matricula = 123;**
 - O **PostgreSQL** pode usar o **índice da PK**?
 - Sim, mas não é ideal.
 - Como a **PK** está ordenada por (nr_livro, matricula, dt_emprestimo), o otimizador só consegue aproveitar o índice começando por nr_livro.
 - Se a busca for apenas por matricula, ele não consegue usar eficientemente esse índice.

Aqui um índice específico em matricula pode trazer ganho real.

Particionamento de Tabelas

- Consultas por nr_livro
- Mesma lógica: **SELECT * FROM movimentacao_2025 WHERE nr_livro = 7;**
- Esse caso já é bem atendido pelo índice da PK (porque nr_livro é a primeira coluna).
- **Não precisa criar índice separado para nr_livro.**

Particionamento de Tabelas

- Consultas por período (**dt_emprestimo**)
- Faz sentido criar **índices específicos em dt_emprestimo** porque nem sempre a busca envolverá as outras colunas da **PK**.
- **Consultas por empréstimos em aberto (dt_efetiva_devolucao IS NULL)**: aqui o índice parcial é interessante, pois nenhum índice atual ajuda.

Particionamento de Tabelas

```
158 CREATE INDEX movimentacao_2025_abertos_idx  
159 ON movimentacao_2025 (matricula, nr_livro)  
160 WHERE dt_efetiva_devolucao IS NULL;  
161
```

Data output Messages

CREATE INDEX

Query returned successfully in 67 msec.

Particionamento de Tabelas

- Aqui cabe uma advertência: **mais índices nem sempre valem a pena!**
- Portanto, não saia criando índices até para consultas que raramente serão executadas.
- Todo INSERT, DELETE ou UPDATE corresponderá a uma atualização dos índices existentes – o que resulta em perda de eficiência.

Particionamento de Tabelas



The screenshot shows a PostgreSQL client interface with the following components:

- Connection Bar:** Displays 'Biblioteca2/postgres@PostgreSQL 14'.
- Toolbar:** Includes icons for file operations, query execution, and settings. A dropdown menu is set to 'No limit'.
- Query Editor:** Contains a single SQL query:

```
1  
2 INSERT INTO MOVIMENTACAO VALUES (5,2,'2025-09-03','2025-09-10',NULL);  
3
```
- Output Panel:** Divided into 'Data output' and 'Messages'. The 'Messages' tab is active, showing:

```
INSERT 0 1  
  
Query returned successfully in 67 msec.
```

Particionamento de Tabelas

Biblioteca2/postgres@PostgreSQL 14

Query Query History

```
1  
2 INSERT INTO MOVIMENTACAO VALUES (5,2,'2025-09-03','2025-09-10',NULL);  
3  
4  
5 SELECT * FROM MOVIMENTACAO;  
6
```

Data output Messages

	nr_livro [PK] integer	matricula [PK] integer	dt_emprestimo [PK] date	dt_prevista_devolucao date	dt_efetiva_devolucao date
1	5	2	2025-09-03	2025-09-10	[null]

Particionamento de Tabelas



The screenshot shows a PostgreSQL client interface with the following components:

- Connection Bar:** Displays 'Biblioteca2/postgres@PostgreSQL 14'.
- Toolbar:** Includes icons for file operations, query execution, and settings. A dropdown menu is set to 'No limit'.
- Query Editor:** Contains the following SQL code:

```
1  
2 INSERT INTO MOVIMENTACAO VALUES (5,2,'2025-09-03','2025-09-10',NULL);  
3  
4  
5 SELECT * FROM MOVIMENTACAO_2025;  
6  
7 -- Repare que o registro não existe no particionamento MOVIMENTACAO_2025.  
8
```
- Data Output Panel:** Shows the results of the query in a table format.

nr_livro	matricula	dt_emprestimo	dt_prevista_devolucao	dt_efetiva_devolucao
integer	integer	date	date	date

```

7  -- Repare que o registro não existe no particionamento MOVIMENTACAO_2025.
8
9  -- Função de roteamento
10 CREATE OR REPLACE FUNCTION movimentacao_insert_trigger()
11 RETURNS TRIGGER AS $$
12 BEGIN
13     IF (NEW.dt_emprestimo >= DATE '2025-01-01' AND NEW.dt_emprestimo <= DATE '2025-12-31') THEN
14         INSERT INTO movimentacao_2025 VALUES (NEW.*);
15
16     ELSIF (NEW.dt_emprestimo >= DATE '2026-01-01' AND NEW.dt_emprestimo <= DATE '2026-12-31') THEN
17         INSERT INTO movimentacao_2026 VALUES (NEW.*);
18
19     ELSIF (NEW.dt_emprestimo >= DATE '2027-01-01' AND NEW.dt_emprestimo <= DATE '2027-12-31') THEN
20         INSERT INTO movimentacao_2027 VALUES (NEW.*);
21
22     ELSE
23         RAISE EXCEPTION 'Data de empréstimo (%) fora do intervalo das partições!', NEW.dt_emprestimo;
24     END IF;
25
26     RETURN NULL; -- impede inserção na tabela mãe
27 END;
28 $$ LANGUAGE plpgsql;
29

```

Data output Messages

CREATE FUNCTION

Query returned successfully in 88 msec.

Particionamento de Tabelas

```
30  
31 CREATE TRIGGER movimentacao_insert_router  
32 BEFORE INSERT ON movimentacao  
33 FOR EACH ROW EXECUTE FUNCTION movimentacao_insert_trigger();  
34
```

Data output Messages

CREATE TRIGGER

Query returned successfully in 74 msec.

Particionamento de Tabelas

```
31 CREATE TRIGGER movimentacao_insert_router
32 BEFORE INSERT ON movimentacao
33 FOR EACH ROW EXECUTE FUNCTION movimentacao_insert_trigger();
34
35 DELETE FROM MOVIMENTACAO WHERE NR_LIVRO = 5 AND MATRICULA = 2
36 AND DT_EMPRESTIMO = '2025-09-03';
37
38 INSERT INTO MOVIMENTACAO VALUES (5,2,'2025-09-03','2025-09-10',NULL);
39
40 SELECT * FROM MOVIMENTACAO_2025;
41
```

Data output Messages








INSERT 0 0

Query returned successfully in 93 msec.

Particionamento de Tabelas

```
30  
31 CREATE TRIGGER movimentacao_insert_router  
32 BEFORE INSERT ON movimentacao  
33 FOR EACH ROW EXECUTE FUNCTION movimentacao_insert_trigger();  
34  
35 DELETE FROM MOVIMENTACAO WHERE NR_LIVRO = 5 AND MATRICULA = 2  
36 AND DT_EMPRESTIMO = '2025-09-03';  
37  
38 INSERT INTO MOVIMENTACAO VALUES (5,2,'2025-09-03','2025-09-10',NULL);  
39  
40 SELECT * FROM MOVIMENTACAO_2025;  
41
```

Data output Messages

      					
	nr_livro integer	matricula integer	dt_emprestimo date	dt_prevista_devolucao date	dt_efetiva_devolucao date
1	5	2	2025-09-03	2025-09-10	[null]

Particionamento de Tabelas

```
30
31 CREATE TRIGGER movimentacao_insert_router
32 BEFORE INSERT ON movimentacao
33 FOR EACH ROW EXECUTE FUNCTION movimentacao_insert_trigger();
34
35 DELETE FROM MOVIMENTACAO WHERE NR_LIVRO = 5 AND MATRICULA = 2
36 AND DT_EMPRESTIMO = '2025-09-03';
37
38 INSERT INTO MOVIMENTACAO VALUES (5,2,'2025-09-03','2025-09-10',NULL);
39
40 SELECT * FROM MOVIMENTACAO_2025;
41
42 SELECT * FROM MOVIMENTACAO_2026;
```

Data output Messages



nr_livro integer	matricula integer	dt_emprestimo date	dt_prevista_devolucao date	dt_efetiva_devolucao date
---------------------	----------------------	-----------------------	-------------------------------	------------------------------

Particionamento de Tabelas

37

38 `INSERT INTO MOVIMENTACAO VALUES (4,1,'2026-01-25','2025-02-01',NULL);`

39

Data output Messages

INSERT 0 0

Query returned successfully in 64 msec.

Particionamento de Tabelas

```
37  
38 INSERT INTO MOVIMENTACAO VALUES (4,1,'2026-01-25','2025-02-01',NULL);  
39  
40 SELECT * FROM MOVIMENTACAO;  
41
```






Data output Messages

	nr_livro [PK] integer	matricula [PK] integer	dt_emprestimo [PK] date	dt_prevista_devolucao date	dt_efetiva_devolucao date
1	5	2	2025-09-03	2025-09-10	[null]
2	4	1	2026-01-25	2025-02-01	[null]

Particionamento de Tabelas

```
37  
38 INSERT INTO MOVIMENTACAO VALUES (4,1,'2026-01-25','2025-02-01',NULL);  
39  
40 SELECT * FROM MOVIMENTACAO;  
41  
42 SELECT * FROM MOVIMENTACAO_2025;  
43
```








Data output Messages

      					
	nr_livro integer	matricula integer	dt_emprestimo date	dt_prevista_devolucao date	dt_efetiva_devolucao date
1	5	2	2025-09-03	2025-09-10	[null]

Particionamento de Tabelas

```
37  
38 INSERT INTO MOVIMENTACAO VALUES (4,1,'2026-01-25','2025-02-01',NULL);  
39  
40 SELECT * FROM MOVIMENTACAO;  
41  
42 SELECT * FROM MOVIMENTACAO_2025;  
43  
44 SELECT * FROM MOVIMENTACAO_2026;  
45
```

Data output Messages

      					
	nr_livro integer	matricula integer	dt_emprestimo date	dt_prevista_devolucao date	dt_efetiva_devolucao date
1	4	1	2026-01-25	2025-02-01	[null]

Particionamento de Tabelas

```
37  
38 INSERT INTO MOVIMENTACAO VALUES (4,1,'2026-01-25','2025-02-01',NULL);  
39  
40 SELECT * FROM MOVIMENTACAO;  
41  
42 SELECT * FROM MOVIMENTACAO_2025;  
43  
44 SELECT * FROM MOVIMENTACAO_2026;  
45  
46 DELETE FROM MOVIMENTACAO WHERE NR_LIVRO = 5 AND MATRICULA = 2  
47 AND DT_EMPRESTIMO = '2025-09-03';  
48
```

Data output Messages

DELETE 1

Query returned successfully in 65 msec.

Particionamento de Tabelas

```
45  
46 DELETE FROM MOVIMENTACAO WHERE NR_LIVRO = 5 AND MATRICULA = 2  
47 AND DT_EMPRESTIMO = '2025-09-03';  
48  
49 SELECT * FROM MOVIMENTACAO; -- EXCLUSÃO BEM SUCEDIDA EM MOVIMENTACAO  
50
```

Data output Messages

	nr_livro [PK] integer	matricula [PK] integer	dt_emprestimo [PK] date	dt_prevista_devolucao date	dt_efetiva_devolucao date
1	4	1	2026-01-25	2025-02-01	[null]

Particionamento de Tabelas

```
45  
46 DELETE FROM MOVIMENTACAO WHERE NR_LIVRO = 5 AND MATRICULA = 2  
47 AND DT_EMPRESTIMO = '2025-09-03';  
48  
49 SELECT * FROM MOVIMENTACAO; -- EXCLUSÃO BEM SUCEDIDA EM MOVIMENTACAO  
50  
51 SELECT * FROM MOVIMENTACAO_2025;  
52
```

Data output Messages

<div><div><div><div><div></div><div></div></div><div><div></div><div></div></div><div><div></div><div></div></div><div><div></div><div></div></div><div><div></div><div></div></div><div><div></div><div></div></div><div><div></div><div></div></div></div></div></div>						
	nr_livro integer		matricula integer	dt_emprestimo date	dt_prevista_devolucao date	dt_efetiva_devolucao date

Particionamento de Tabelas

52

53 `SELECT * FROM MOVIMENTACAO_2026;`

54

Data output Messages



	nr_livro integer	matricula integer	dt_emprestimo date	dt_prevista_devolucao date	dt_efetiva_devolucao date
1	4	1	2026-01-25	2025-02-01	[null]

Particionamento de Tabelas

```
52  
53 SELECT * FROM MOVIMENTACAO_2026;  
54  
55 UPDATE MOVIMENTACAO SET DT_PREVISTA_DEVOLUCAO = '2026-02-01'  
56 WHERE NR_LIVRO = 4 AND MATRICULA = 1 AND DT_EMPRESTIMO = '2026-01-25';  
57
```

Data output Messages

UPDATE 1

Query returned successfully in 83 msec.

Particionamento de Tabelas

```
52
53 SELECT * FROM MOVIMENTACAO_2026;
54
55 UPDATE MOVIMENTACAO SET DT_PREVISTA_DEVOLUCAO = '2026-02-01'
56 WHERE NR_LIVRO = 4 AND MATRICULA = 1 AND DT_EMPRESTIMO = '2026-01-25';
57
58 SELECT * FROM MOVIMENTACAO_2026; -- Alteração com sucesso em MOVIMENTACAO_2026
59
```

Data output Messages



	nr_livro integer	matricula integer	dt_emprestimo date	dt_prevista_devolucao date	dt_efetiva_devolucao date
1	4	1	2026-01-25	2026-02-01	[null]

Particionamento de Tabelas

- Uma maneira alternativa de Particionar Tabelas.
- Inicialmente apagamos as tabelas recém criadas no particionamento.

```
59
60 DROP TABLE MOVIMENTACAO_2025;
61 DROP TABLE MOVIMENTACAO_2026;
62 DROP TABLE MOVIMENTACAO_2027;
63
```

Data output	Messages
DROP TABLE	
Query returned successfully in 105 msec.	

Particionamento de Tabelas

```
64 DROP TABLE MOVIMENTACAO;
65
66 CREATE TABLE MOVIMENTACAO (
67     NR_LIVRO INTEGER NOT NULL,
68     MATRICULA INTEGER NOT NULL,
69     DT_EMPRESTIMO DATE NOT NULL CHECK (DT_EMPRESTIMO > '1996-03-25') DEFAULT CURRENT_DATE,
70     DT_PREVISTA_DEVOLUCAO DATE CHECK (DT_PREVISTA_DEVOLUCAO > DT_EMPRESTIMO),
71     DT_EFETIVA_DEVOLUCAO DATE CHECK (DT_EFETIVA_DEVOLUCAO IS NULL OR DT_EFETIVA_DEVOLUCAO >= DT_EMPRESTIMO)
72
73     -- RETIRE POIS NÃO FUNCIONA NA MANEIRA COMO FAREMOS A PARTIÇÃO
74     --PRIMARY KEY(NR_LIVRO, MATRICULA, DT_EMPRESTIMO),
75
76     --FOREIGN KEY (NR_LIVRO) REFERENCES LIVRO (NR_LIVRO)
77     --ON DELETE RESTRICT,
78
79     --FOREIGN KEY (MATRICULA) REFERENCES USUARIO(MATRICULA)
80     --ON DELETE RESTRICT
81 ) PARTITION BY RANGE (DT_EMPRESTIMO);
82
```

Data output Messages

CREATE TABLE

Query returned successfully in 51 msec.

Particionamento de Tabelas

```
79      --FOREIGN KEY (MATRICULA) REFERENCES USUARIO(MATRICULA)
80      --ON DELETE RESTRICT
81  ) PARTITION BY RANGE (DT_EMPRESTIMO);
82
83  CREATE TABLE MOVIMENTACAO_2025 PARTITION OF MOVIMENTACAO
84  FOR VALUES FROM ('2025-01-01') TO ('2026-01-01');
85
86  -- A primeira data ('2025-01-01') é INCLUSIVE mas a
87  -- segunda data do intervalo ('2026-01-01') é EXCLUSIVE.
88  -- PERÍODO ABRANGIDO: de 01-01-2025 até 31-12-2025.
89
```

Data output Messages

CREATE TABLE

Query returned successfully in 51 msec.

Particionamento de Tabelas

```
89  
90 CREATE TABLE MOVIMENTACAO_2026 PARTITION OF MOVIMENTACAO  
91 FOR VALUES FROM ('2026-01-01') TO ('2027-01-01');  
92  
93 CREATE TABLE MOVIMENTACAO_2027 PARTITION OF MOVIMENTACAO  
94 FOR VALUES FROM ('2027-01-01') TO ('2028-01-01');  
95  
96 CREATE TABLE MOVIMENTACAO_2028 PARTITION OF MOVIMENTACAO  
97 FOR VALUES FROM ('2028-01-01') TO ('2029-01-01');  
98  
99  
100
```

Data output Messages

CREATE TABLE

Query returned successfully in 81 msec.

Particionamento de Tabelas

```
89  
90 CREATE TABLE MOVIMENTACAO_2026 PARTITION OF MOVIMENTACAO  
91 FOR VALUES FROM ('2026-01-01') TO ('2027-01-01');  
92  
93 CREATE TABLE MOVIMENTACAO_2027 PARTITION OF MOVIMENTACAO  
94 FOR VALUES FROM ('2027-01-01') TO ('2028-01-01');  
95  
96 CREATE TABLE MOVIMENTACAO_2028 PARTITION OF MOVIMENTACAO  
97 FOR VALUES FROM ('2028-01-01') TO ('2029-01-01');  
98  
99 INSERT INTO MOVIMENTACAO VALUES (6,3,NOW()::DATE,(NOW() + INTERVAL '7 DAYS'), NULL);  
100
```

Data output Messages

INSERT 0 1

Query returned successfully in 83 msec.

Particionamento de Tabelas

```
95  
96 CREATE TABLE MOVIMENTACAO_2028 PARTITION OF MOVIMENTACAO  
97 FOR VALUES FROM ('2028-01-01') TO ('2029-01-01');  
98  
99 INSERT INTO MOVIMENTACAO VALUES (6,3,NOW()::DATE,(NOW() + INTERVAL '7 DAYS'), NULL);  
100  
101 SELECT * FROM MOVIMENTACAO;  
102
```

Data output Messages



	nr_livro integer	matricula integer	dt_emprestimo date	dt_prevista_devolucao date	dt_efetiva_devolucao date
1	6	3	2025-09-03	2025-09-10	[null]

Particionamento de Tabelas

```
98
99 INSERT INTO MOVIMENTACAO VALUES (6,3,NOW()::DATE,(NOW() + INTERVAL '7 DAYS'), NULL);
100
101 SELECT * FROM MOVIMENTACAO;
102
103 SELECT * FROM MOVIMENTACAO_2025; -- NÃO FOI NECESSÁRIO CRIAR TRIGGER
104
```

Data output Messages



	nr_livro integer	matricula integer	dt_emprestimo date	dt_prevista_devolucao date	dt_efetiva_devolucao date
1	6	3	2025-09-03	2025-09-10	[null]

Particionamento de Tabelas

```
95  
96 CREATE TABLE MOVIMENTACAO_2028 PARTITION OF MOVIMENTACAO  
97 FOR VALUES FROM ('2028-01-01') TO ('2029-01-01');  
98  
99 INSERT INTO MOVIMENTACAO VALUES (6,3,NOW()::DATE,(NOW() + INTERVAL '7 DAYS'), NULL);  
100  
101 SELECT * FROM MOVIMENTACAO;  
102  
103 SELECT * FROM MOVIMENTACAO_2025; -- NÃO FOI NECESSÁRIO CRIAR TRIGGER  
104  
105 SELECT * FROM MOVIMENTACAO_2026;
```

Data output Messages



nr_livro	matricula	dt_emprestimo	dt_prevista_devolucao	dt_efetiva_devolucao
integer	integer	date	date	date

Particionamento de Tabelas

- Criar partições em tabelas do **PostgreSQL** (seja com **herança** ou com o **particionamento nativo PARTITION BY** a partir da **versão 10**) traz várias vantagens importantes, principalmente em cenários de **grandes volumes de dados**.

Particionamento de Tabelas

- 1. Melhora de desempenho em consultas:
- O **PostgreSQL** aplica **partition pruning**: ele só acessa a partição relevante em vez de varrer a tabela inteira.
- **Exemplo:** buscar empréstimos de 2025 acessa apenas MOVIMENTACAO_2025.
- Menos páginas lidas implica em consultas mais rápidas.

Particionamento de Tabelas

- **Partition pruning:** É o processo pelo qual o otimizador de consultas do PostgreSQL decide quais partições realmente precisam ser acessadas em uma consulta.
- **Ou seja:** em vez de varrer todas as tabelas-filhas, ele “poda” (descarta) aquelas que não têm chance de conter os dados pedidos.

Particionamento de Tabelas

- Suponha que você tenha a tabela movimentacao particionada por ano:
 - **movimentacao_2025**
 - **movimentacao_2026**
 - **movimentacao_2027**
- Agora você executa:
- **SELECT ***
- **FROM movimentacao**
- **WHERE dt_emprestimo BETWEEN '2025-01-01' AND '2025-12-31';**

Particionamento de Tabelas

- Com **partition pruning**, o **PostgreSQL** olha para a condição e só consulta a partição **movimentacao_2025**, ignorando as demais (2026, 2027).
- Sem pruning, ele teria que verificar todas as partições e só depois descartar as linhas que não atendem ao filtro (processo muito mais custoso).

Particionamento de Tabelas

- Outra vantagem do Particionamento: Índices menores e mais eficientes
- Cada partição tem seus índices locais.
- Em vez de ter um índice gigante sobre milhões de linhas, você terá vários índices menores e mais rápidos de percorrer.
- Isso reduz também o custo de manutenção dos índices (inserções, updates).

Particionamento de Tabelas

- Outra vantagem do Particionamento:
- **Manutenção simplificada**
- Operações como **VACUUM**, **ANALYZE** e **REINDEX** podem ser feitas em partições específicas, sem bloquear a tabela inteira.
- Isso ajuda na **administração de bases grandes em produção**.

Particionamento de Tabelas

- Outra vantagem do Particionamento:
- **Arquivamento e limpeza de dados**
- É fácil descartar ou arquivar dados antigos com **DROP TABLE** ou **DETACH PARTITION**, sem precisar rodar um **DELETE** massivo.
- **Exemplo: para remover empréstimos de 2025, basta dropar a tabela MOVIMENTACAO_2025.**

Particionamento de Tabelas

- Outra **vantagem do Particionamento:**
- **Paralelismo natural**
- Consultas que envolvem várias partições podem ser executadas em paralelo.
- Isso aproveita melhor servidores com múltiplos núcleos.

Particionamento de Tabelas

- Outra vantagem do Particionamento:
- **Facilidade de organização lógica**
- Os dados ficam separados por período, região, categoria, etc.
- Ajuda na clareza do modelo e no planejamento de crescimento do banco.

Particionamento de Tabelas

- **Quando vale a pena particionar?**
 - Quando a tabela tem muitos milhões de registros.
 - Quando as consultas mais comuns filtram naturalmente por um critério de partição (ex.: datas, regiões, tipos).
 - Quando há necessidade de arquivamento periódico (dados de anos passados, logs, movimentações históricas).

Particionamento de Tabelas

- Em nosso exemplo, como a tabela MOVIMENTACAO tende a crescer ano a ano, o particionamento por dt_emprestimo é mais do que indicado:
- Consultas do tipo "empréstimos de 2025" serão bem mais rápidas.
- Registros antigos podem ser descartados facilmente.