



CEFET/RJ



ANÁLISE DE PLANOS DE EXECUÇÃO DE CONSULTAS NOS SBGD

Eduardo Ogasawara
eogasawara@ieee.org
<https://eic.cefet-rj.br/~eogasawara>

Esquema de análise

- DEPARTAMENTO
 - PRIMARY KEY (DNUMERO)
- EMPREGADO
 - PRIMARY KEY (CPF)
 - FOREIGN KEY (GERENTECPF) REFERENCES EMPREGADO (CPF),
 - FOREIGN KEY (DNO) REFERENCES DEPARTAMENTO (DNUMERO)
- DEPT_LOCALIZACOES
 - PRIMARY KEY (DNUMERO,DLOCALIZACAO),
 - FOREIGN KEY (DNUMERO) REFERENCES DEPARTAMENTO (DNUMERO)
- PROJETO
 - PRIMARY KEY (PNUMERO)
 - FOREIGN KEY (DNUM) REFERENCES DEPARTAMENTO (DNUMERO)
- TRABALHA_EM
 - PRIMARY KEY (ECPF,PNO)
 - FOREIGN KEY (ECPF) REFERENCES EMPREGADO (CPF),
 - FOREIGN KEY (PNO) REFERENCES PROJETO (PNUMERO)
- DEPENDENTE
 - PRIMARY KEY (ECPF, NOME_DEPENDENTE),
 - FOREIGN KEY (ECPF) REFERENCES EMPREGADO (CPF)

PostgreSQL

Consultas pela chave primária (empregado a partir do cpf)

The screenshot shows the pgAdmin 4 web interface. On the left, the 'Browser' pane displays a tree structure of database objects, with 'projetos' selected. The main pane is divided into two sections: the 'Query Editor' and the 'Query History'. The 'Query Editor' contains the following SQL query:

```
1 SELECT * FROM EMPREGADO E WHERE E.CPF = '999887777'
```

Below the query editor, the 'Data Output' tab is active, showing the 'Explain' view. The 'Explain' view displays the execution plan for the query. The plan consists of a single node, a 'Seq Scan on empregado as e', with a cost of 0..1.35, 1 row, and a width of 78. The filter applied is '((cpf)::bpchar = '999887777'::bpchar)'. The 'Rows' column shows 1 row.

| # | Node | Rows |
|----|--|------|
| 1. | → Seq Scan on empregado as e (cost=0..1.35 rows=1 width=78) Filter: ((cpf)::bpchar = '999887777'::bpchar) | 1 |

SELECT * FROM EMPREGADO E WHERE E.CPF = '999887777'

Consulta por valor (empregado a partir do primeiro nome)

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays a tree structure of database objects. The 'projetos' database is selected, showing its contents: Casts, Catalogs, Event Triggers, Extensions, Foreign Data Wrappers, Languages, Publications, Schemas, Subscriptions, Login/Group Roles, and Tablespaces. The main pane is divided into several tabs: Dashboard, Properties, SQL, Statistics, Dependencies, and Dependents. The 'SQL' tab is active, showing a query editor with the following SQL statement:

```
1 SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'
```

Below the query editor, the 'Data Output' tab is active, displaying the execution plan. The plan shows a single node: a Seq Scan on empregado as e, with a cost of 0..1.35, 1 row, and a width of 78. The filter is ((pname)::text = 'Alicia'::text). The plan is visualized as a tree structure with a single node.

| # | Node | Rows |
|----|---|------|
| 1. | → Seq Scan on empregado as e (cost=0..1.35 rows=1 width=78) Filter: ((pname)::text = 'Alicia'::text) | 1 |

SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'

Sub-consulta na projeção

(lista de departamentos com seus respectivos gerentes)

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane shows a tree of database objects, with 'projetos' selected. The main pane displays a SQL query in the 'Query Editor' tab:

```

1 SELECT D.*,
2     (SELECT PNOME FROM EMPREGADO E WHERE E.CPF = D.GERCPF) AS PNOME
3 FROM DEPARTAMENTO D
4
5

```

Below the query editor, the 'Explain' tab is active, showing the execution plan. The plan consists of two nodes:

| # | Node | Rows |
|----|--|------|
| 1. | → Seq Scan on departamento as d (cost=0..5.08 rows=3 width=117) | 3 |
| 2. | → Seq Scan on empregado as e (cost=0..1.35 rows=1 width=8) Filter: ((cpf)::bpchar = (d.gercpf)::bpchar) | 1 |

```

SELECT D.*,
       (SELECT PNOME FROM EMPREGADO E WHERE E.CPF = D.GERCPF) AS PNOME
FROM DEPARTAMENTO D

```

Consulta equivalente em junção (lista de departamentos com seus respectivos gerentes)

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane shows a tree of database objects, with 'projetos' selected. The main pane displays a SQL query in the 'Query Editor' tab:

```
1 SELECT D.*, E.PNOME
2 FROM DEPARTAMENTO D, EMPREGADO E
3 WHERE E.CPF = D.GERCPF
4
5
```

Below the query editor, the 'Explain' tab is active, showing the execution plan. The plan consists of four nodes:

| # | Node | Rows |
|----|--|------|
| 1. | → Hash Inner Join (cost=1.07..2.48 rows=3 width=47) Hash Cond: ((e.cpf)::bpchar = (d.gercpf)::bpchar) | 3 |
| 2. | → Seq Scan on empregado as e (cost=0..1.28 rows=28 width=20) | 28 |
| 3. | → Hash (cost=1.03..1.03 rows=3 width=39) | 3 |
| 4. | → Seq Scan on departamento as d (cost=0..1.03 rows=3 width=39) | 3 |

```
SELECT D.*, E.PNOME
FROM DEPARTAMENTO D, EMPREGADO E
WHERE E.CPF = D.GERCPF
```

Árvore de consulta equivalente em junção (lista de departamentos com seus respectivos gerentes)

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays a tree structure of database objects, with 'projetos' selected. The main pane shows the 'Query Editor' with the following SQL query:

```
1 SELECT D.*, E.PNOME
2 FROM DEPARTAMENTO D, EMPREGADO E
3 WHERE E.CPF = D.GERCPF
4
5
```

Below the query editor, the 'Explain' tab is active, displaying the 'Graphical' query plan. The plan shows two input tables, 'departamento' and 'empregado', each represented by a grid icon. Arrows from both tables point to a 'Hash' icon, which then points to a 'Hash Inner Join' icon, illustrating the execution strategy for the join operation.

```
SELECT D.*, E.PNOME
FROM DEPARTAMENTO D, EMPREGADO E
WHERE E.CPF = D.GERCPF
```


Subconsulta não correlacionada com in (empregados com dependentes)

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane shows a tree view of the database structure, with 'projetos' selected. The main pane displays a SQL query in the 'Query Editor' tab:

```
1 SELECT *
2 FROM EMPREGADO E
3 WHERE E.CPF IN (SELECT D.ECPF FROM DEPENDENTE D)
4
5
```

Below the query editor, the 'Explain' tab is active, showing the execution plan. The plan consists of five steps:

| # | Node | Rows |
|----|--|------|
| 1. | → Hash Inner Join (cost=8.61..10.21 rows=23 width=78) Hash Cond: ((e.cpf)::bpchar = (d.ecpf)::bpchar) | 23 |
| 2. | → Seq Scan on empregado as e (cost=0..1.28 rows=28 width=78) | 28 |
| 3. | → Hash (cost=8.32..8.32 rows=23 width=12) | 23 |
| 4. | → Aggregate (cost=8.09..8.32 rows=23 width=12) | 23 |
| 5. | → Seq Scan on dependente as d (cost=0..7.07 rows=407 width=12) | 407 |

```
SELECT *
FROM EMPREGADO E
WHERE E.CPF IN (SELECT D.ECPF FROM DEPENDENTE D)
```

Subconsulta correlacionada com in (empregados com dependentes do mesmo sexo)

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane shows a tree structure with 'projetos' expanded. The main pane is divided into 'Query Editor' and 'Query History'. The 'Query Editor' contains the following SQL query:

```
1 SELECT *
2 FROM EMPREGADO E
3 WHERE E.CPF IN (SELECT D.ECPF FROM DEPENDENTE D WHERE D.SEXO = E.SEXO)
4
5
```

Below the query editor, the 'Explain' tab is selected, showing the execution plan. The plan consists of two nodes:

| # | Node | Rows |
|----|---|------|
| 1. | → Seq Scan on empregado as e (cost=0..114.72 rows=14 width=78) Filter: (SubPlan 1) | 14 |
| 2. | → Seq Scan on dependente as d (cost=0..8.09 rows=4 width=12) Filter: ((sexo)::text = (e.sexo)::text) | 4 |

```
SELECT *
FROM EMPREGADO E
WHERE E.CPF IN (SELECT D.ECPF FROM DEPENDENTE D WHERE D.SEXO = E.SEXO)
```

Subconsulta correlacionada com exists (empregados com dependentes do mesmo sexo)

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane shows a tree structure with 'projetos' expanded. The 'Query Editor' pane contains the following SQL query:

```
1 SELECT *
2 FROM EMPREGADO E
3 WHERE EXISTS (SELECT * FROM DEPENDENTE D WHERE D.ECPF = E.CPF AND D.SEXO = E.SEXO)
4
5
```

Below the query editor, the 'Explain' tab is selected, showing the execution plan. The plan consists of five nodes:

| # | Node | Rows |
|----|--|------|
| 1. | → Hash Inner Join (cost=10.13..11.64 rows=7 width=78) Hash Cond: (((e.cpf)::bpchar = (d.ecpf)::bpchar) AND ((e.sexo)::text = (d.sexo)::text)) | 7 |
| 2. | → Seq Scan on empregado as e (cost=0..1.28 rows=28 width=78) | 28 |
| 3. | → Hash (cost=9.52..9.52 rows=41 width=14) | 41 |
| 4. | → Aggregate (cost=9.11..9.52 rows=41 width=14) | 41 |
| 5. | → Seq Scan on dependente as d (cost=0..7.07 rows=407 width=14) | 407 |

```
SELECT *
FROM EMPREGADO E
WHERE EXISTS (SELECT * FROM DEPENDENTE D WHERE D.ECPF = E.CPF AND D.SEXO = E.SEXO)
```

Consulta com agregação

(quantidade de empregados que trabalham em mais de um projeto)

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane shows a tree of database objects, with 'projetos' selected. The main pane displays a SQL query in the 'Query Editor' tab:

```
1 SELECT E.PNOME, COUNT(*) AS QTD
2 FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
3 GROUP BY E.PNOME HAVING COUNT(*) > 1
4
5
```

Below the query editor, the 'Data Output' tab is active, showing the 'Explain' view. The execution plan is displayed in a table with columns '#', 'Node', and 'ROWS'. The plan consists of five steps:

| # | Node | ROWS |
|----|---|------|
| 1. | → Aggregate (cost=2.92..3.12 rows=5 width=16) Filter: (count(*) > 1) | 5 |
| 2. | → Hash Inner Join (cost=1.63..2.84 rows=16 width=8) Hash Cond: ((te.ecpf)::bpchar = (e.cpf)::bpchar) | 16 |
| 3. | → Seq Scan on trabalha_em as te (cost=0..1.16 rows=16 width=12) | 16 |
| 4. | → Hash (cost=1.28..1.28 rows=28 width=20) | 28 |
| 5. | → Seq Scan on empregado as e (cost=0..1.28 rows=28 width=20) | 28 |

```
SELECT E.PNOME, COUNT(*) AS QTD
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
GROUP BY E.PNOME HAVING COUNT(*) > 1
```

Criação de índice

The screenshot displays the pgAdmin 4 web interface. On the left, the 'Browser' pane shows a tree of database objects, with 'projetos' selected. The main area is divided into several tabs: 'Dashboard', 'Properties', 'SQL', 'Statistics', 'Dependencies', and 'Dependents'. The 'SQL' tab is active, showing a 'Query Editor' with the following SQL command:

```
1 CREATE INDEX EMP_PNAME_I ON EMPREGADO(PNAME)
2
3
4
5
```

Below the query editor, the 'Messages' tab is selected, displaying the output: 'CREATE INDEX' and 'Query returned successfully in 120 msec.'.

In the bottom right corner, a 'Maintenance' popup window is visible, indicating a successful VACUUM operation on the 'projetos' database. The window shows the command 'VACUUM (FULL, FREEZE, VERBOSE) on database 'projetos' of server 127.0.0.1 (127.0.0.1:5432)' was executed on 'Thu Nov 21 2024 00:12:52 GMT-0300 (Brasilia Standard Time)' and took '13.88 seconds'. A green bar at the bottom of the popup states 'Successfully completed.'.

CREATE INDEX EMP_PNAME_I ON EMPREGADO(PNAME)

Índice nem sempre é usado (busca pelo empregado a partir do primeiro nome)

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays a tree of database objects, with 'projetos' selected. The main pane is divided into several tabs: 'Query Editor', 'Query History', 'Data Output', 'Explain', 'Messages', 'Notifications', 'Graphical', 'Analysis', and 'Statistics'. The 'Query Editor' tab is active, showing the following SQL query:

```
1 SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'
```

Below the query editor, the 'Explain' tab is active, displaying the execution plan. The plan shows a single node: a sequential scan on the 'empregado' table, filtered by the condition '((pnome)::text = 'Alicia')::text'. The cost is 0..1.35, with 1 row and a width of 78.

| # | Node | Rows |
|----|---|------|
| 1. | → Seq Scan on empregado as e (cost=0..1.35 rows=1 width=78) Filter: ((pnome)::text = 'Alicia')::text | 1 |

SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'

MySQL

Consultas pela chave primária (empregado a partir do cpf)

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar contains the Navigator pane with a Schemas view showing a tree structure for 'grupo9' and 'projetos'. The 'projetos' schema is expanded, showing tables: DEPARTAMENTO, DEPENDENTE, DEPT_LOCALIZACOE, EMPREGADO, PROJETO, and TRABALHA_EM. The main query editor displays the following SQL query:

```
1 • EXPLAIN ANALYZE SELECT * FROM EMPREGADO E WHERE E.CPF = '999887777'
```

The query is executed, and the bottom pane shows the 'Form Editor' tab with the following output:

```
EXPLAIN:
-> Rows fetched before execution (cost=0..0 rows=1) (actual time=188e-6..292e-6 rows=1 loops=1)
```

The bottom status bar indicates 'Query Completed'.

SELECT * FROM EMPREGADO E WHERE E.CPF = '999887777'

Consulta por valor (empregado a partir do primeiro nome)

The screenshot shows the MySQL Workbench interface. On the left, the 'Schemas' pane shows the 'projetos' database selected, with a list of tables including DEPARTAMENTO, DEPENDENTE, DEPT_LOCALIZACOES, EMPREGADO, PROJETO, and TRABALHA_EM. The main query editor displays the following SQL query:

```
1 EXPLAIN ANALYZE SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'
```

Below the query editor, the 'Form Editor' tab shows the 'EXPLAIN' output:

```
EXPLAIN:
-> Filter: (E.PNOME = 'Alicia') (cost=3.05 rows=2.8) (actual time=0.111..0.114 rows=1 loops=1)
-> Table scan on E (cost=3.05 rows=28) (actual time=0.0432..0.103 rows=28 loops=1)
```

The right sidebar contains buttons for 'Result Grid', 'Form Editor', and 'Field Types'. The bottom status bar indicates 'Read Only'.

SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'

Sub-consulta na projeção

(lista de departamentos com seus respectivos gerentes)

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar contains a Navigator pane with a SCHEMAS section, a search bar, and a tree view showing the database structure. The main query editor displays a query with line numbers 1 through 7. The bottom pane shows the execution plan (EXPLAIN) for the query, detailing the cost, rows, and loops for each step. The status bar at the bottom indicates 'Query Completed'.

MySQL Workbench

127.0.0.1 x

File Edit View Query Database Server Tools Scripting Help

Navigator queries x

SCHEMAS

Filter objects

grupo9

projetos

Tables

- DEPARTAMENTO
- DEPENDENTE
- DEPT_LOCALIZACOE
- EMPREGADO
- PROJETO
- TRABALHA_EM

Administration Schemas

Information

No object selected

Object Info Session

Query Completed

1 • EXPLAIN ANALYZE SELECT D.*,

2 (SELECT PNAME FROM EMPREGADO E WHERE E.CPF = D.GERCPF) AS PNAME

3 FROM DEPARTAMENTO D

4

5

6

7

Limit to 1000 rows

EXPLAIN:

- > Table scan on D (cost=0.55 rows=3) (actual time=0.0233..0.0286 rows=3 loops=1)
- > Select #2 (subquery in projection; dependent)
- > Single-row index lookup on E using PRIMARY (CPF=D.GERCPF) (cost=0.35 rows=1) (actual time=0.00927..0.00937 rows=1 loops=3)

Result Grid

Form Editor

Read Only

```
SELECT D.*,  
  (SELECT PNAME FROM EMPREGADO E WHERE E.CPF = D.GERCPF) AS PNAME  
FROM DEPARTAMENTO D
```

Consulta com agregação

(quantidade de empregados que trabalham em mais de um projeto)

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' panel with a tree view showing 'grupo9' and 'projetos' (containing tables like DEPARTAMENTO, EMPREGADO, PROJETO, and TRABALHA_EM). The main editor shows a query in the 'queries' tab:

```
1 • EXPLAIN ANALYZE SELECT E.PNOME, COUNT(*) AS QTD
2 FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
3 GROUP BY E.PNOME HAVING COUNT(*) > 1
4
5
6
7
8
```

Below the query, the 'Form Editor' tab shows the execution plan (EXPLAIN) results:

```
EXPLAIN:
-> Filter: ('count(0)' > 1) (actual time=0.178..0.182 rows=6 loops=1)
-> Table scan on <temporary> (actual time=0.175..0.178 rows=8 loops=1)
-> Aggregate using temporary table (actual time=0.172..0.172 rows=8 loops=1)
-> Nested loop inner join (cost=7.45 rows=16) (actual time=0.0474..0.116 rows=16 loops=1)
```

The bottom status bar indicates 'Query Completed'.

```
SELECT E.PNOME, COUNT(*) AS QTD
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
GROUP BY E.PNOME HAVING COUNT(*) > 1
```

Junção com busca por valor (último nome e horas trabalhadas por projeto pelo primeiro nome)

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows a database named 'projetos' with several tables, including 'TRABALHA_EM'. The main editor window displays a SQL query:

```
1 • EXPLAIN ANALYZE SELECT E.UNOME, TE.HORAS
2 FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
3 WHERE E.PNOME = 'Alicia'
4
5
```

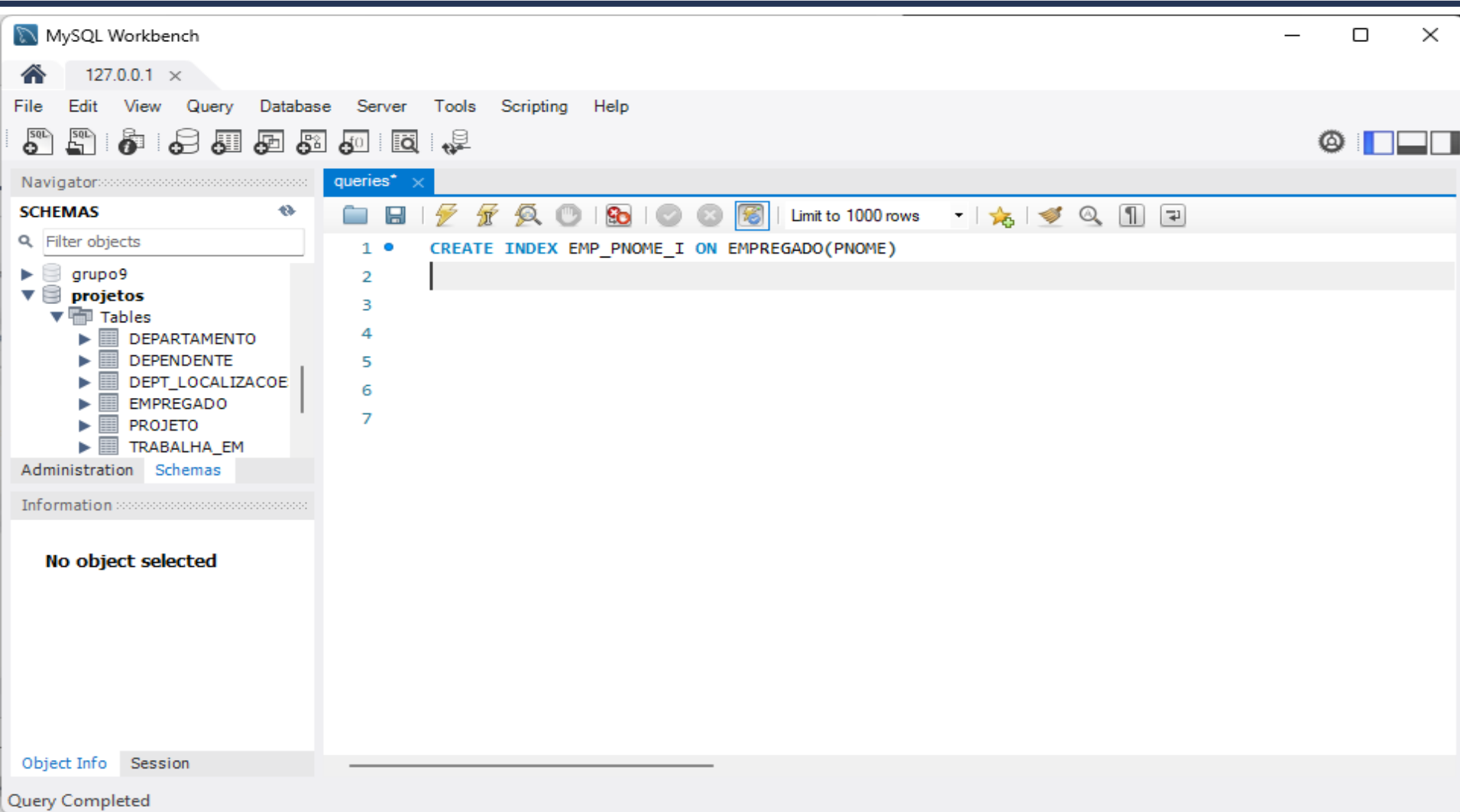
Below the query, the 'EXPLAIN' output is shown in the 'Form Editor' tab:

```
EXPLAIN:
-> Nested loop inner join (cost=5.01 rows=5.6) (actual time=0.0983..0.104 rows=2 loops=1)
-> Filter: (E.PNOME = 'Alicia') (cost=3.05 rows=2.8) (actual time=0.0726..0.0743 rows=1 loops=1)
-> Table scan on E (cost=3.05 rows=28) (actual time=0.042..0.0653 rows=28 loops=1)
-> Index lookup on TE using PRIMARY (ECPF=E.CPF) (cost=0.571 rows=2) (actual time=0.0229..0.026 rows=2)
```

The bottom status bar indicates 'Query Completed'.

```
SELECT E.UNOME, TE.HORAS
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
WHERE E.PNOME = 'Alicia'
```

Criação de índice



CREATE INDEX EMP_PNOME_I ON EMPREGADO(PNOME)

Uso tradicional do índice (busca pelo empregado a partir do primeiro nome)

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows a database named 'projeto9' with a table 'EMPREGADO' selected. The main query editor displays the following SQL query:

```
EXPLAIN ANALYZE SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'
```

The query has been executed, and the 'EXPLAIN' output is shown in the 'Form Editor' pane. The output indicates that an index lookup was performed on the 'EMP_PNOME_I' index for the value 'Alicia'.

```
EXPLAIN:
-> Index lookup on E using EMP_PNOME_I (PNOME='Alicia') (cost=0.35 rows=1) (actual time=0.0443..0.0511 rows=1 loops=1)
```

The status bar at the bottom indicates 'Query Completed'.

SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'

Junção tradicional

(índice do nome do empregado foi usado)

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows a database named 'projetos' with several tables, including 'EMPREGADO' and 'TRABALHA_EM'. The main editor displays a query in the 'Form Editor' tab:

```
1 • EXPLAIN ANALYZE SELECT E.UNOME, TE.HORAS
2 FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
3 WHERE E.PNOME = 'Alicia'
```

The 'EXPLAIN' output is shown in the 'Result Grid' tab, indicating a nested loop inner join with an index lookup on the 'EMPREGADO' table using the 'EMP_PNOME_I' index for the filter 'Alicia'.

EXPLAIN:

- > Nested loop inner join (cost=1.05 rows=2) (actual time=0.0634..0.0712 rows=2 loops=1)
- > Index lookup on E using EMP_PNOME_I (PNOME='Alicia') (cost=0.35 rows=1) (actual time=0.0432..0.0456 rows=1 loops=1)
- > Index lookup on TE using PRIMARY (ECPF=E.CPF) (cost=0.7 rows=2) (actual time=0.0179..0.0225 rows=2)

```
SELECT E.UNOME, TE.HORAS
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
WHERE E.PNOME = 'Alicia'
```

SQL Server

Consulta por valor (empregado a partir do primeiro nome)

The screenshot displays the Microsoft SQL Server Management Studio interface. The main window shows a query in the SQLQuery1.sql file: `SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'`. The query has been executed successfully, as indicated by the status bar at the bottom. The execution plan for the query is shown in the lower-left pane, indicating a **Clustered Index Scan (Clustered)** on the `[EMPREGADO].[PK__EMPREGAD__C1F89730...]` index, with a cost of 100%.

The **Properties** pane on the right shows the execution details for the `SELECT` operation. The **Estimated Subtree Cost** is 0,0033128. The **Estimated Number of Rows** is 3,5. The **Estimated Operator Cost** is 0 (0%).

| Property | Value |
|-------------------------------|------------------|
| Cached plan size | 24 KB |
| CardinalityEstimationMod | 160 |
| CompileCPU | 0 |
| CompileMemory | 136 |
| CompileTime | 0 |
| Estimated Number of Row | 0 |
| Estimated Number of Row | 3,5 |
| Estimated Operator Cost | 0 (0%) |
| Estimated Subtree Cost | 0,0033128 |
| MemoryGrantInfo | |
| Optimization Level | TRIVIAL |
| OptimizerHardwareDepen | |
| OptimizerStatsUsage | |

Query 1: Query cost (relative to the batch): 100%
SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'

Query executed successfully. | 127.0.0.1 (16.0 RTM) | sa (70) | projetos | 00:00:00 | 0 rows

SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'

Sub-consulta na projeção

(lista de departamentos com seus respectivos gerentes)

The screenshot displays the Microsoft SQL Server Management Studio interface. The title bar indicates the connection is 'SQLQuery1.sql - 127.0.0.1.projeto (sa (70))'. The menu bar includes File, Edit, View, Project, Tools, Window, and Help. The toolbar contains various icons for file operations, query execution, and formatting. The 'projeto' database is selected in the server explorer.

The query editor shows the following SQL query:

```
SELECT D.*,  
       (SELECT PNOME FROM EMPREGADO E WHERE E.CPF = D.GERCPF) AS PNOME  
FROM DEPARTAMENTO D
```

The 'Execution plan' tab is active, showing the query cost (100%) and the execution plan diagram. The plan consists of the following steps:

- SELECT** (Cost: 0 %)
- Compute Scalar** (Cost: 0 %)
- Nested Loops (Left Outer Join)** (Cost: 0 %)
- Clustered Index Scan (Clustered)** [DEPARTAMENTO].[PK_DEPARTAM_FAE85...] (Cost: 48 %)
- Clustered Index Seek (Clustered)** [EMPREGADO].[PK_EMPREGAD_C1F89730...] (Cost: 52 %)

The 'Properties' pane on the right shows the following details for the 'SELECT' operation:

| Properties | |
|---|------------------|
| SELECT | |
| Misc | |
| Cached plan size | 24 KB |
| CardinalityEstimationMod | 160 |
| CompileCPU | 3 |
| CompileMemory | 232 |
| CompileTime | 3 |
| Estimated Number of Row | 0 |
| Estimated Number of Row | 3 |
| Estimated Operator Cost | 0 (0%) |
| Estimated Subtree Cost | 0,0068974 |
| MemoryGrantInfo | |
| Optimization Level | FULL |
| OptimizerHardwareDepen | |
| OptimizerStatsUsage | |
| Estimated Subtree Cost | |
| Estimated cumulative cost of this operation and all child operations. | |

The status bar at the bottom indicates 'Query executed successfully.' and '127.0.0.1 (16.0 RTM) | sa (70) | projeto | 00:00:00 | 0 rows'.

```
SELECT D.*,  
       (SELECT PNOME FROM EMPREGADO E WHERE E.CPF = D.GERCPF) AS PNOME  
FROM DEPARTAMENTO D
```

Consulta equivalente em junção (lista de departamentos com seus respectivos gerentes)

SQLQuery1.sql - 127.0.0.1.projeto (sa (70))* - Microsoft SQL Server Management Studio

File Edit View Project Tools Window Help

projeto Execute

SQLQuery1.sql - 127.0.0.1.projeto (sa (70))*

```
SELECT D.*, E.PNOME
FROM DEPARTAMENTO D, EMPREGADO E
WHERE E.CPF = D.GERCPF
```

100 %

Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

SELECT D.*, E.PNOME FROM DEPARTAMENTO D, EMPREGADO E WHERE E.CPF = D.GERCPF

Execution Plan:

- SELECT (Cost: 0 %)
- Nested Loops (Inner Join) (Cost: 0 %)
 - Clustered Index Scan (Clustered) [DEPARTAMENTO].[PK_DEPARTAM_FAE85...] (Cost: 48 %)
 - Clustered Index Seek (Clustered) [EMPREGADO].[PK_EMPREGAD_C1F89730...] (Cost: 52 %)

Query executed successfully. | 127.0.0.1 (16.0 RTM) | sa (70) | projeto | 00:00:00 | 0 rows

Ready

Properties

SELECT

Misc

| | |
|--------------------------|-----------|
| Cached plan size | 24 KB |
| CardinalityEstimationMod | 160 |
| CompileCPU | 2 |
| CompileMemory | 200 |
| CompileTime | 2 |
| Estimated Number of Row | 0 |
| Estimated Number of Row | 3 |
| Estimated Operator Cost | 0 (0%) |
| Estimated Subtree Cost | 0,0068971 |
| MemoryGrantInfo | |
| Optimization Level | FULL |
| OptimizerHardwareDepen | |
| OptimizerStatsUsage | |

Estimated Subtree Cost

Estimated cumulative cost of this operation and all child operations.

```
SELECT D.*, E.PNOME
FROM DEPARTAMENTO D, EMPREGADO E
WHERE E.CPF = D.GERCPF
```

Subconsulta não correlacionada com in (empregados com dependentes)

SQLQuery1.sql - 127.0.0.1.projeto (sa (70)) - Microsoft SQL Server Management Studio

File Edit View Project Tools Window Help

projeto Execute

SQLQuery1.sql - 127.0.0.1.projeto (sa (70))

```
SELECT *  
FROM EMPREGADO E  
WHERE E.CPF IN (SELECT D.ECPF FROM DEPENDENTE D)
```

100 %

Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

SELECT * FROM EMPREGADO E WHERE E.CPF IN (SELECT D.ECPF FROM DEPENDENTE D)

SELECT Cost: 0 %

Nested Loops (Left Semi Join) Cost: 1 %

Clustered Index Scan (Clustered) [EMPREGADO].[PK_EMPREGADO_C1F89730...] Cost: 30 %

Clustered Index Seek (Clustered) [DEPENDENTE].[PK_DEPENDENTE_2BB4A8D...] Cost: 69 %

Query executed successfully. | 127.0.0.1 (16.0 RTM) | sa (70) | projeto | 00:00:00 | 0 rows

Ready

Properties

SELECT

Misc

| | |
|--------------------------|-----------|
| Cached plan size | 24 KB |
| CardinalityEstimationMod | 160 |
| CompileCPU | 6 |
| CompileMemory | 288 |
| CompileTime | 6 |
| Estimated Number of Row | 0 |
| Estimated Number of Row | 3 |
| Estimated Operator Cost | 0 (0%) |
| Estimated Subtree Cost | 0,0109816 |
| MemoryGrantInfo | |
| Optimization Level | FULL |
| OptimizerHardwareDepen | |
| OptimizerStatsUsage | |

Estimated Subtree Cost

Estimated cumulative cost of this operation and all child operations.

```
SELECT *  
FROM EMPREGADO E  
WHERE E.CPF IN (SELECT D.ECPF FROM DEPENDENTE D)
```

Subconsulta correlacionada com exists (empregados com dependentes do mesmo sexo)

SQLQuery1.sql - 127.0.0.1.projeto (sa (70))* - Microsoft SQL Server Management Studio

File Edit View Project Tools Window Help

projeto Execute

SQLQuery1.sql - 127....1.projeto (sa (70))*

```
SELECT *  
FROM EMPREGADO E  
WHERE EXISTS (SELECT * FROM DEPENDENTE D WHERE D.ECPF = E.CPF AND D.SEXO = E.SEXO)
```

100 %

Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

SELECT * FROM EMPREGADO E WHERE EXISTS (SELECT * FROM DEPENDENTE D WHERE D.ECPF...

SELECT
Cost: 0 %

Nested Loops
(Left Semi Join)
Cost: 1 %

Clustered Index Scan (Clustered)
[EMPREGADO].[PK_EMPREGAD_C1F89730...]
Cost: 30 %

Clustered Index Seek (Clustered)
[DEPENDENTE].[PK_DEPENDEN_2BB4A8D...]
Cost: 69 %

Query executed successfully. | 127.0.0.1 (16.0 RTM) | sa (70) | projeto | 00:00:00 | 0 rows

Ready

Properties

SELECT

Misc

| | |
|--------------------------|-----------|
| Cached plan size | 32 KB |
| CardinalityEstimationMod | 160 |
| CompileCPU | 7 |
| CompileMemory | 336 |
| CompileTime | 7 |
| Estimated Number of Row | 0 |
| Estimated Number of Row | 28 |
| Estimated Operator Cost | 0 (0%) |
| Estimated Subtree Cost | 0,0109951 |
| MemoryGrantInfo | |
| Optimization Level | FULL |
| OptimizerHardwareDepen | |
| OptimizerStatsUsage | |

Estimated Subtree Cost

Estimated cumulative cost of this operation and all child operations.

```
SELECT *  
FROM EMPREGADO E  
WHERE EXISTS (SELECT * FROM DEPENDENTE D WHERE D.ECPF = E.CPF AND D.SEXO = E.SEXO)
```

Consulta com agregação

(quantidade de empregados que trabalham em mais de um projeto)

SQLQuery1.sql - 127.0.0.1.projeto (sa (70))* - Microsoft SQL Server Management Studio

File Edit View Project Tools Window Help

projeto Execute

```
SELECT E.PNOME, COUNT(*) AS QTD
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
GROUP BY E.PNOME HAVING COUNT(*) > 1
```

100 %

Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

SELECT E.PNOME, COUNT(*) AS QTD FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = ...

Query executed successfully. | 127.0.0.1 (16.0 RTM) | sa (70) | projeto | 00:00:00 | 0 rows

Ready

Properties

SELECT

Misc

| | |
|--------------------------|----------|
| Cached plan size | 32 KB |
| CardinalityEstimationMod | 160 |
| CompileCPU | 4 |
| CompileMemory | 296 |
| CompileTime | 5 |
| Estimated Number of Row | 0 |
| Estimated Number of Row | 2,28486 |
| Estimated Operator Cost | 0 (0%) |
| Estimated Subtree Cost | 0,020494 |

MemoryGrantInfo

Optimization Level FULL

OptimizerHardwareDepen

OptimizerStatsUsage

Estimated Subtree Cost

Estimated cumulative cost of this operation and all child operations.

```
SELECT E.PNOME, COUNT(*) AS QTD
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
GROUP BY E.PNOME HAVING COUNT(*) > 1
```

Junção com busca por valor (último nome e horas trabalhadas por projeto pelo primeiro nome)

The screenshot displays the Microsoft SQL Server Management Studio interface. The query editor shows the following SQL query:

```
SELECT E.UNOME, TE.HORAS
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
WHERE E.PNOME = 'Alicia'
```

The execution plan for the query is shown below the editor. It consists of a 'SELECT' operator (Cost: 0 %) connected to a 'Nested Loops (Inner Join)' operator (Cost: 1 %). The 'Nested Loops' operator is connected to two 'Clustered Index Scan (Clustered)' operators. The first scan is on the 'EMPREGADO' table (Cost: 47 %) and the second is on the 'TRABALHA_EM' table (Cost: 52 %).

The 'Properties' window on the right shows the following values:

| SELECT | |
|---|-----------|
| Misc | |
| Cached plan size | 24 KB |
| CardinalityEstimationMod | 160 |
| CompileCPU | 3 |
| CompileMemory | 240 |
| CompileTime | 3 |
| Estimated Number of Row | 0 |
| Estimated Number of Row | 7 |
| Estimated Operator Cost | 0 (0%) |
| Estimated Subtree Cost | 0,0070377 |
| MemoryGrantInfo | |
| Optimization Level | FULL |
| OptimizerHardwareDepen | |
| OptimizerStatsUsage | |
| Estimated Subtree Cost | |
| Estimated cumulative cost of this operation and all child operations. | |

The status bar at the bottom indicates: Query executed successfully. | 127.0.0.1 (16.0 RTM) | sa (70) | projetos | 00:00:00 | 0 rows

```
SELECT E.UNOME, TE.HORAS
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
WHERE E.PNOME = 'Alicia'
```

Índice de cobertura de consulta

- O índice de cobertura de consulta possibilita processar a consulta sem selecionar a tabela
- Todos os atributos usados na consulta para uma Tabela T estão presentes no índice
- A ordem de criação dos campos no índice é relevante
 - Primeiro os atributos de seleção (mais restritivos primeiro)
 - Segundo os atributos usados na junção
 - Terceiro os atributos usados na projeção

Índice de cobertura de consulta

The screenshot displays the Microsoft SQL Server Management Studio interface. The main query window shows the following SQL command:

```
CREATE INDEX EMP_COVER ON EMPREGADO(PNOME) INCLUDE (CPF, UNOME)
```

The Messages pane below the query window indicates that the commands were completed successfully. The completion time is 2024-11-21T08:09:50.8609699-03:00.

The Properties pane on the right shows the current connection parameters for the connection named 127.0.0.1 (sa).

| Aggregate Status | |
|---------------------|---------------------|
| Connection failures | |
| Elapsed time | 00:00:00.3570063 |
| Finish time | 21/11/2024 08:09:50 |
| Name | 127.0.0.1 |
| Rows returned | 0 |
| Start time | 21/11/2024 08:09:50 |
| State | Open |

| Connection | |
|-----------------|----------------|
| Connection name | 127.0.0.1 (sa) |

| Connection Details | |
|-------------------------|---------------------|
| Connection elapsed time | 00:00:00.3570063 |
| Connection encryption | Not encrypted |
| Connection finish time | 21/11/2024 08:09:50 |

Name
The name of the connection.

The status bar at the bottom shows the query was executed successfully, with 0 rows returned. The connection details are 127.0.0.1 (16.0 RTM) | sa (70) | projetos | 00:00:00 | 0 rows.

CREATE INDEX EMP_COVER ON EMPREGADO(PNOME) INCLUDE (CPF, UNOME)

Uso do índice de cobertura de consulta (Consulta de alocação por data de nascimento com índice)

SQLQuery1.sql - 127.0.0.1,1433.projeto (sa (58))* - Microsoft SQL Server Management Studio

File Edit View Project Tools Window Help

projeto Execute

```
SELECT E.UNOME, TE.HORAS
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
WHERE E.PNOME = 'Alicia'
```

100 %

Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

SELECT E.UNOME, TE.HORAS FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF WHE...

SELECT Cost: 0

Nested Loops (Inner Join) Cost: 0

Index Seek (NonClustered) [EMPREGADO].[EMP_COVER] [E] Cost: 50

Clustered Index Seek (Clustered) [TRABALHA_EM].[PK_TRABALHA_EM] Cost: 50

Query executed successfully. | 127.0.0.1,1433 (16.0 RTM) | sa (58) | projeto | 00:00:00 | 0 rows

Ready

Properties

SELECT

Misc

| | |
|---------------------------|-------------------|
| Cached plan size | 24 KB |
| CardinalityEstimate | 160 |
| CompileCPU | 3 |
| CompileMemory | 248 |
| CompileTime | 3 |
| Estimated Number 0 | |
| Estimated Number 2 | |
| Estimated Operator 0 (0%) | |
| Estimated Subtree | 0,0065757 |
| MemoryGrantInfo | |
| Optimization Level | FULL |
| OptimizerHardware | |
| OptimizerStatsUsage | |
| QueryHash | 0x86586179216741C |

Estimated Subtree Cost

Estimated cumulative cost of this operation...

```
SELECT E.UNOME, TE.HORAS
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
WHERE E.PNOME = 'Alicia'
```

Oracle

Consultas pela chave primária (empregado a partir do cpf)

The screenshot shows the Oracle SQL Developer interface with the following components:

- Worksheet:** Contains the SQL query: `SELECT * FROM EMPREGADO E WHERE E.CPF = '999887777'`
- Query Result:** Shows the execution plan for the query.

Execution Plan Table:

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------|-------------|----------------|-------------|------|
| SELECT STATEMENT | | | 1 | 1 |
| TABLE ACCESS | EMPREGADO | BY INDEX ROWID | 1 | 1 |
| INDEX | SYS_C004766 | UNIQUE SCAN | 1 | 1 |

Access Predicates:

- E.CPF='999887777'

Other XML:

- {info}
- info type="db_version"
10.2.0.1
- info type="parse_schema"

SELECT * FROM EMPREGADO E WHERE E.CPF = '999887777'

Consulta por valor (empregado a partir do primeiro nome)

The screenshot shows the Oracle SQL Developer interface. The main window displays a query in the Worksheet: `SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'`. Below the query, the Query Result tab is active, showing the execution plan. The plan details are as follows:

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------|-------------|---------|-------------|------|
| SELECT STATEMENT | | | 1 | 3 |
| TABLE ACCESS | EMPREGADO | FULL | 1 | 3 |

Additional details in the plan include Filter Predicates: `E.PNOME='Alicia'`, and Other XML information: `{info}` with `info type="db_version" 10.2.0.1` and `info type="parse_schema" "PROJETOS"`.

SELECT * FROM EMPREGADO E WHERE E.PNOME = 'Alicia'

Sub-consulta na projeção

(lista de departamentos com seus respectivos gerentes)

The screenshot shows the Oracle SQL Developer interface. The Query Builder tab is active, displaying the following SQL query:

```
SELECT D.*,  
       (SELECT PNAME FROM EMPREGADO E WHERE E.CPF = D.GERCPF) AS PNAME  
FROM DEPARTAMENTO D
```

Below the query, the 'Query Result' tab is selected, showing the execution plan. The plan details are as follows:

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------------|--------------|----------------|-------------|------|
| SELECT STATEMENT | | | 3 | 3 |
| TABLE ACCESS | EMPREGADO | BY INDEX ROWID | 1 | 1 |
| INDEX | SYS_C004766 | UNIQUE SCAN | 1 | 1 |
| Access Predicates | | | | |
| E.CPF=:B1 | | | | |
| TABLE ACCESS | DEPARTAMENTO | FULL | 3 | 3 |
| Other XML | | | | |
| {info} | | | | |
| info type="db_version" | | | | |
| 10.2.0.1 | | | | |

```
SELECT D.*,  
       (SELECT PNAME FROM EMPREGADO E WHERE E.CPF = D.GERCPF) AS PNAME  
FROM DEPARTAMENTO D
```

Consulta equivalente em junção (lista de departamentos com seus respectivos gerentes)

The screenshot shows the Oracle SQL Developer interface. The main window displays a query in the Query Builder:

```
SELECT D.*, E.PNOME  
FROM DEPARTAMENTO D, EMPREGADO E  
WHERE E.CPF = D.GERCPF
```

Below the query, the Query Result tab is active, showing the execution plan. The plan details are as follows:

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------|--------------|----------------|-------------|------|
| SELECT STATEMENT | | | 3 | 6 |
| NESTED LOOPS | | | 3 | 6 |
| TABLE ACCESS | DEPARTAMENTO | FULL | 3 | 3 |
| TABLE ACCESS | EMPREGADO | BY INDEX ROWID | 1 | 1 |
| INDEX | SYS_C004766 | UNIQUE SCAN | 1 | 0 |

The execution plan also includes a diagram on the left showing the flow of data and the access predicates: `E.CPF=D.GERCPF`. The status bar at the bottom indicates the cursor is at Line 4 Column 1.

```
SELECT D.*, E.PNOME  
FROM DEPARTAMENTO D, EMPREGADO E  
WHERE E.CPF = D.GERCPF
```

Subconsulta não correlacionada com in (empregados com dependentes)

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main window has a tab for 'projetos' and a 'Query Builder' tab. The query editor contains the following SQL code:

```
FROM EMPREGADO E
WHERE E.CPF IN (SELECT D.ECPF FROM DEPENDENTE D)
```

Below the query editor, the 'Query Result' tab is active, showing the execution plan. The plan is as follows:

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|-----------------------------------|-------------|------------|-------------|------|
| SELECT STATEMENT | | | 23 | 7 |
| HASH JOIN | | RIGHT SEMI | 23 | 7 |
| Access Predicates E.CPF=D.ECPF | | | | |
| TABLE ACCESS | DEPENDENTE | FULL | 407 | 3 |
| TABLE ACCESS | EMPREGADO | FULL | 28 | 3 |

The bottom status bar shows 'Line 4 Column 1 | Insert | Modified | Windows: CF'.

```
SELECT *
FROM EMPREGADO E
WHERE E.CPF IN (SELECT D.ECPF FROM DEPENDENTE D)
```


Subconsulta correlacionada com exists (empregados com dependentes do mesmo sexo)

The screenshot shows the Oracle SQL Developer interface. The main window displays a query in the 'Query Builder' tab:

```
SELECT *  
FROM EMPREGADO E  
WHERE EXISTS (SELECT * FROM DEPENDENTE D WHERE D.ECPF = E.CPF AND D.SEXO = E.SEXO)
```

Below the query, the 'Query Result' tab is active, showing the execution plan. The plan is a 'HASH JOIN' with 'Access Predicates' and 'AND' conditions. The 'TABLE ACCESS' for 'DEPENDENTE' and 'EMPREGADO' are both 'FULL'.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|-------------------|-------------|------------|-------------|------|
| SELECT STATEMENT | | | 1 | 7 |
| HASH JOIN | | RIGHT SEMI | 1 | 7 |
| Access Predicates | | | | |
| AND | | | | |
| D.ECPF=E.CPF | | | | |
| D.SEXO=E.SEXO | | | | |
| TABLE ACCESS | DEPENDENTE | FULL | 407 | 3 |
| TABLE ACCESS | EMPREGADO | FULL | 28 | 3 |

```
SELECT *  
FROM EMPREGADO E  
WHERE EXISTS (SELECT * FROM DEPENDENTE D WHERE D.ECPF = E.CPF AND D.SEXO = E.SEXO)
```

Consulta com agregação

(quantidade de empregados que trabalham em mais de um projeto)

The screenshot shows the Oracle SQL Developer interface. The main window displays a query in the 'Query Builder' tab:

```
SELECT E.PNOME, COUNT(*) AS QTD
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
GROUP BY E.PNOME HAVING COUNT(*) > 1
```

Below the query, the 'Query Result' tab shows the execution plan. The plan is as follows:

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|-------------------|-------------|------------|-------------|------|
| SELECT STATEMENT | | | 16 | 6 |
| FILTER | | | | |
| Filter Predicates | | | | |
| COUNT(*)>1 | | | | |
| HASH | | | | |
| NESTED LOOPS | | GROUP BY | 16 | 6 |
| TABLE ACCESS | | | 16 | 5 |
| EMPREGADO | | FULL | 28 | 3 |
| INDEX | SYS_C004778 | RANGE SCAN | 1 | 1 |
| Access Predicates | | | | |
| TE.ECPF=E.CPF | | | | |

```
SELECT E.PNOME, COUNT(*) AS QTD
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF
GROUP BY E.PNOME HAVING COUNT(*) > 1
```

Junção com busca por valor (último nome e horas trabalhadas por projeto pelo primeiro nome)

The screenshot shows the Oracle SQL Developer interface. The main window displays a query in the Query Builder:

```
SELECT E.UNOME, TE.HORAS  
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF  
WHERE E.PNOME = 'Alicia'
```

Below the query, the Execution Plan is shown. It includes a tree view on the left and a table on the right.

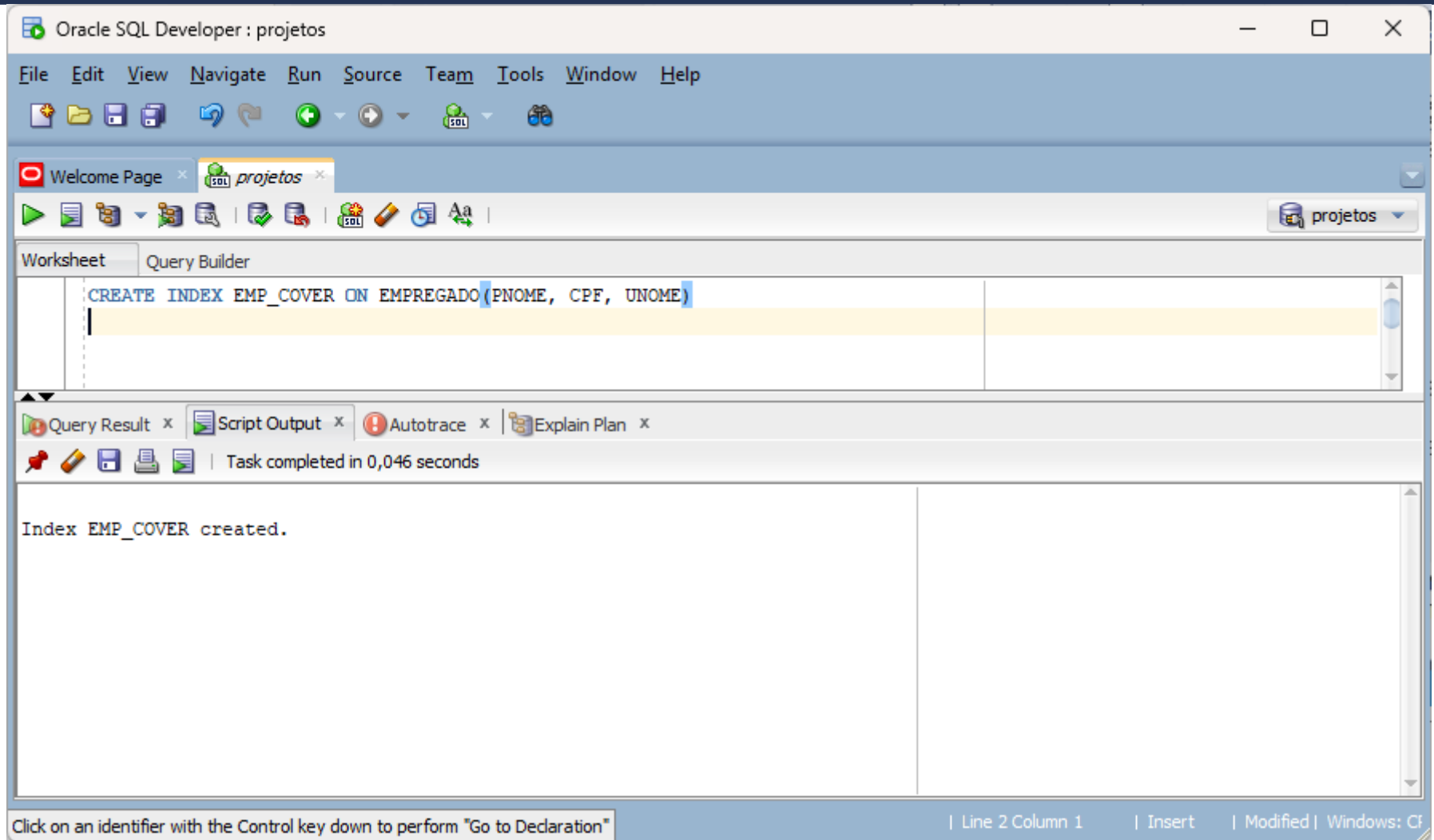
| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|---|-------------|---------|-------------|------|
| SELECT STATEMENT | | | 2 | 7 |
| HASH JOIN | | | 2 | 7 |
| Access Predicates TE.ECPF=E.CPF | | | | |
| TABLE ACCESS Filter Predicates E.PNOME='Alicia' | EMPREGADO | FULL | 1 | 3 |
| TABLE ACCESS | TRABALHA_EM | FULL | 16 | 3 |
| Other XML {info} | | | | |

```
SELECT E.UNOME, TE.HORAS  
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF  
WHERE E.PNOME = 'Alicia'
```

Índice de cobertura de consulta

- O índice de cobertura de consulta possibilita processar a consulta sem selecionar a tabela
- Todos os atributos usados na consulta para uma Tabela T estão presentes no índice
- A ordem de criação dos campos no índice é relevante
 - Primeiro os atributos de seleção (mais restritivos primeiro)
 - Segundo os atributos usados na junção
 - Terceiro os atributos usados na projeção

Índice de cobertura de consulta



CREATE INDEX EMP_COVER ON EMPREGADO(PNOME, CPF, UNOME)

Uso do índice de cobertura de consulta

(Consulta de alocação por data de nascimento com índice)

The screenshot displays the Oracle SQL Developer interface. The 'Connections' pane on the left shows a tree view of database objects for 'BD@localhost'. The main window shows a SQL query in the 'Query Builder' tab:

```
SELECT E.UNOME, TE.HORAS  
FROM EMPREGADO E JOIN TRABALHA_EM TE ON E.CPF = TE.ECPF  
WHERE E.PNOME = 'Alicia'
```

Below the query, the 'Explain Plan' tab is active, showing the execution plan for the query. The plan is as follows:

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------|-------------|------------|-------------|------|
| SELECT STATEMENT | | | 2 | 4 |
| NESTED LOOPS | | | 2 | 4 |
| TABLE ACCESS | TRABALHA_EM | FULL | 16 | 3 |
| INDEX | EMP_COVER | RANGE SCAN | 1 | 1 |

The execution plan diagram shows the following structure:

- SELECT STATEMENT
 - NESTED LOOPS
 - TABLE ACCESS (TRABALHA_EM)
 - INDEX (EMP_COVER) with Access Predicates:
 - AND
 - E.PNOME='Alicia'
 - TE.ECPF=E.CPF

The status bar at the bottom indicates 'Line 4 Column 1 | Insert | Modified | Windows: CF'.

Referências

