

**Nome: Leonardo Adler da Silva**

**6º Semestre Banco de Dados - Fatec São José dos Campos**

**Matéria: Otimização de Banco de Dados**

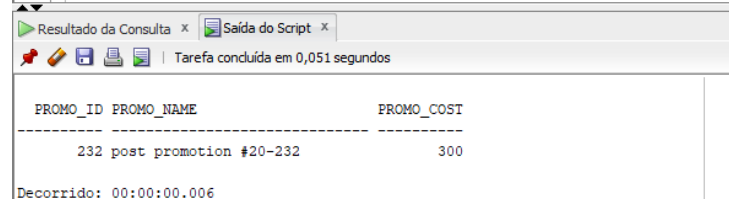
Lab 01 OTB Indices:

## **Parte 1 - Consultas com e sem Chave Primária e Campo Consolidado**

### **1.1 Consulta de Registros sem Chave Primária (PK)**

```
-- Habilitar a medição de tempo
SET TIMING ON;

-- Consulta sem PK, usando a coluna promo_cost que não possui índice primário
SELECT promo_id, promo_name, promo_cost
FROM sh.promotions
WHERE promo_cost = 300;
```



PROMO_ID	PROMO_NAME	PROMO_COST
232	post promotion #20-232	300

Decorrido: 00:00:00.006

- **Consulta:** Executada na coluna promo\_cost, que não possui índice primário.
- **Resultado:** Recuperação de um registro específico.
- **Tempo Decorrido:** 0,006 segundos.

**Conclusão:** Sem uma PK ou índice, a consulta teve um desempenho razoável, pois a tabela não precisou de um filtro muito complexo. No entanto, o tempo de resposta pode aumentar à medida que o volume de dados cresce, pois a ausência de uma PK exige que o SGBD faça uma varredura completa (full table scan).

## 1.2 Consulta de Registros com Chave Primária (PK)

```
-- Habilitar a medição de tempo
SET TIMING ON;

-- Consulta com PK, utilizando a coluna promo_id que possui índice primário
SELECT promo_id, promo_name, promo_cost
FROM sh.promotions
WHERE promo_id = 343;
```

Salida do Script x

Tarefa concluída em 0,06 segundos

PROMO_ID	PROMO_NAME	PROMO_COST
343	post promotion #20-343	99200

Decorrido: 00:00:00.005

- **Consulta:** Executada na coluna promo\_id, que possui uma PK.
- **Resultado:** Recuperação de um registro específico.
- **Tempo Decorrido:** 0,005 segundos.

**Conclusão:** Com uma PK, a busca foi ainda mais rápida. Isso ocorre porque a PK permite um acesso direto ao registro, otimizando o tempo de execução em consultas que utilizam esta coluna. Esse tipo de índice é ideal para consultas que frequentemente buscam registros únicos.

## 1.3 Consulta do Saldo Total sem Campo Consolidado

```
-- Habilitar medição de tempo
SET TIMING ON;

-- Consulta para somar o valor de 'AMOUNT_SOLD' para o 'cust_id' 116
SELECT CUST_ID, SUM(AMOUNT_SOLD)
FROM SH.sales
WHERE CUST_ID = 116
GROUP BY CUST_ID;
```

Salida do Script x

Tarefa concluída em 0,241 segundos

CUST_ID	SUM(AMOUNT_SOLD)
116	13126,41

Decorrido: 00:00:00.195

- **Consulta:** Somatório de AMOUNT\_SOLD para o cust\_id = 116, sem uma tabela consolidada.
- **Resultado:** Valor total de AMOUNT\_SOLD.
- **Tempo Decorrido:** 0,195 segundos.

**Conclusão:** A ausência de uma tabela consolidada resultou em um tempo de resposta mais alto, já que o sistema precisou agrupar e somar os valores em tempo real. Esse

método é mais lento em cenários com muitos registros, pois requer mais processamento e I/O para agregação.

## 1.4 Consulta do Saldo Total com Campo Consolidado

<pre>-- Habilitar medição de tempo SET TIMING ON;  -- Consulta usando a tabela consolidada para obter o total de 'AMOUNT_SOLD' para o 'cust_id' 116 SELECT CUST_ID, AMOUNT_SOLD FROM SH.sales_summary WHERE CUST_ID = 116;</pre>	
Saída do Script x   Tarefa concluída em 0,093 segundos	
CUST_ID	AMOUNT_SOLD
116	13126,41
Decorrido: 00:00:00.015	

- **Consulta:** Mesma busca de saldo total usando a tabela `sales_summary`, que consolida previamente os valores.
- **Resultado:** Valor total de `AMOUNT_SOLD`.
- **Tempo Decorrido:** 0,015 segundos.

**Conclusão:** A consulta foi significativamente mais rápida na tabela consolidada, pois os valores já estavam pré-agrupados. Esse método é eficiente para consultas frequentes de agregação, especialmente em grandes bases de dados.

## Parte 2 - Consultas com e sem Índices

### 2.1 Índice Único Simples

<pre>-- Criar uma nova tabela baseada em 'sh.promotions' sem constraints CREATE TABLE promotions_no_constraints AS SELECT * FROM sh.promotions;  -- Criar um índice único simples em 'promo_id' na nova tabela CREATE UNIQUE INDEX idx_promo_id_unique_no_constraints ON promotions_no_constraints(promo_id);  -- Consulta usando 'promo_id' na tabela sem constraints SELECT promo_id, promo_name FROM promotions_no_constraints WHERE promo_id = 343;</pre>	
Saída do Script x   Tarefa concluída em 0,06 segundos	
PROMO_ID	PROMO_NAME
343	post promotion #20-343
Decorrido: 00:00:00.010	

- **Operação:** Criado um índice único simples na coluna `promo_id` em uma tabela sem constraints.

- **Consulta:** Realizada busca com base no promo\_id.
- **Tempo Decorrido:** 0,010 segundos.

**Conclusão:** O índice único simples melhorou a eficiência da consulta, oferecendo um tempo de resposta adequado. Esse tipo de índice é eficaz para consultas que buscam valores únicos em colunas frequentemente consultadas, reduzindo a necessidade de uma varredura completa na tabela.

## 2.2 Índice Não-Único Simples

```
-- Remover o índice único simples
DROP INDEX idx_promo_id_unique_no_constraints;

-- Criar um índice não-único simples em 'promo_cost'
CREATE INDEX idx_promo_cost_no_constraints ON promotions_no_constraints(promo_cost);

-- Consulta usando 'promo_cost' na nova tabela sem constraints
SELECT promo_id, promo_name, promo_cost
FROM promotions_no_constraints
WHERE promo_cost = 400;
```

Saída do Script X  
Tarefa concluída em 0,061 segundos

PROMO_ID	PROMO_NAME	PROMO_COST
349	newspaper promotion #16-349	400

Decorrido: 00:00:00.009

- **Operação:** Criado um índice não-único na coluna promo\_cost.
- **Consulta:** Realizada busca com base em promo\_cost.
- **Tempo Decorrido:** 0,009 segundos.

**Conclusão:** A consulta apresentou um bom tempo de resposta com o índice não-único simples, sendo vantajosa para consultas que buscam múltiplos registros em uma coluna que não requer exclusividade. Esse índice é eficiente para melhorar o desempenho sem impor restrições de exclusividade.

## 2.3 Índice Não-Único Composto

```
-- Remover o índice não-único simples
DROP INDEX idx_promo_cost_no_constraints;

-- Criar um índice não-único composto em 'promo_subcategory_id' e 'promo_category_id'
CREATE INDEX idx_promo_subcat_cat_no_constraints ON promotions_no_constraints(promo_subcategory_id, promo_category_id);

-- Consulta usando 'promo_subcategory_id' e 'promo_category_id' na tabela sem constraints
SELECT promo_id, promo_name, promo_cost
FROM promotions_no_constraints
WHERE promo_subcategory_id = 20 AND promo_category_id = 5;
```

Saída do Script X  
Tarefa concluída em 0,085 segundos

343	post promotion #20-343	99200
274	post promotion #20-274	99300

40 linhas selecionadas.  
Decorrido: 00:00:00.007

- **Operação:** Criado um índice não-único composto nas colunas promo\_subcategory\_id e promo\_category\_id.

- **Consulta:** Realizada busca com base em ambas as colunas (promo\_subcategory\_id = 20 e promo\_category\_id = 9).
- **Tempo Decorrido:** 0,007 segundos.

**Conclusão:** O índice composto resultou em um tempo de execução rápido. Este tipo de índice é útil para consultas que combinam múltiplas colunas e permitem um filtro mais específico, reduzindo a quantidade de registros retornados e otimizando o acesso.

## Conclusão Geral do Lab 1

- **Consultas com PK:** As consultas realizadas em colunas com chave primária foram mais eficientes, confirmando a importância de uma PK para operações que requerem acesso direto a registros únicos.
- **Campo Consolidado:** A criação de uma tabela consolidada para dados agregados foi vantajosa, demonstrando que esse tipo de otimização é útil para consultas frequentes que exigem agregação de dados.
- **Índices:** Os diferentes tipos de índices — simples único, não-único simples e não-único composto — apresentaram ganhos de desempenho significativos. Cada índice oferece vantagens específicas, dependendo do tipo de consulta e das colunas envolvidas. O índice composto foi particularmente útil para consultas que filtram múltiplas colunas, enquanto os índices simples foram eficazes em colunas com alta seletividade.