**RELATÓRIO DE ANALISE**

**TUNNING E PERFORMANCE.**

**CONSULTA ANALISADA – QUERY4.SQL**

|  |
| --- |
|  |

Execução inicial em 974 msec para 17 linhas afetadas;

Analisando o plano de execução:

|  |
| --- |
|  |

**LAUDO -** Essa consulta busca a quantidade de apostas por status. Se a tabela tiver uma quantidade grande de campos, por mais que seja uma contagem, a função COUNT(\*) vai provisionar em memória todos os campos e seu metadados.

Sobre o plano, identificamos que:

1. O fitro PLACED\_AT >= NOW() - INTERVAL '1 HOUR' não encontrou registros dentro da janela da última hora, então todo o esforço de leitura (PARALLEL SEQ SCAN) não produziu nenhum resultado.
2. Identificamos um **PARALLEL SEQ SCAN (TABLE SCAN)**, ou seja, a tabela está sendo lida por inteiro sem nenhum tipo de índice.

**SUGESTÃO DE MELHORIAS**

Criar índice baseado em data e com isso, permitindo que o PostgreSQL leia apenas o intervalo da última hora, reduzindo leituras desnecessárias.

**CREATE INDEX IDX\_BETS\_PLACED\_AT ON BETS (PLACED\_AT);**

E prevendo que possa haver muitos status, e que se for comum consultar pelos campos **PLACED\_AT** e agrupar por **STATUS**.

**CREATE INDEX IDX\_BETS\_PLACED\_AT\_STATUS ON BETS (PLACED\_AT, STATUS);**

**APÓS A APLICAÇÃO DAS SUGESTÕES**

Com a aplicação das sugestões, o plano de execução trouxe:

|  |
| --- |
|  |

* O uso do índice idx\_bets\_placed\_at\_status permitiu ao banco realizar um Index Only Scan, usando apenas o índice sem precisar acessar a tabela (Heap). Isso é ideal para consultas que só precisam das colunas indexadas.
* Houve a redução drástica do tempo de execução. De ~850 ms caiu para aproximadamente 0.2 ms, o que significa um ganho de performance de cerca de 4.250x.
* As ausências de leituras desnecessárias da tabela foram eliminadas, economizando muitos I/O e CPU.

Concluindo, a consulta continua correta, agregando status e contando apostas da última hora, agora de forma muito mais rápida.