



# **Projeto Kali - Documentação final Base de Dados – SQL**

Autor: Edson Monteiro de Almeida

Github: [https://github.com/edsmont/kali\\_projeto\\_sql\\_coder](https://github.com/edsmont/kali_projeto_sql_coder)

Instituição: CODERHOUSE

Treinamento: SQL

Professor: Levi Cruz

Data: 23/02/2024

## 1. Introdução:

Este projeto tem como objetivo modelar um banco de dados relacional para centralizar e integrar informações de diversas ferramentas de monitoração (Zabbix, Grafana, Orion, SIEM), sistema de inventário e sistema ITSM, com foco em auxiliar o time de operação de uma empresa que oferece serviços de telecomunicações, cloud e segurança. A criação dessa base de dados visa solucionar os desafios da gestão de dados em silos, proporcionando uma visão unificada do ambiente de TI e facilitando a análise e correlação de informações relevantes para a operação dos serviços, resolvendo os desafios da gestão de dados distribuídos e melhorando a correlação de informações críticas.

## 2. Objetivo:

O objetivo principal é construir um modelo de banco de dados robusto e escalável, capaz de armazenar e organizar dados de diferentes fontes para a aplicação **Kali**, que está sendo desenvolvida com foco na **otimização da gestão da infraestrutura de TI** e na **garantia da qualidade dos serviços de telecomunicações, cloud e segurança**. O modelo de dados permitirá:

- **Controlar a disponibilidade do ambiente e serviços**, facilitando a identificação de falhas e a tomada de ações proativas.
- **Relacionar eventos de monitoração com seus respectivos ativos e contratos**, fornecendo informações relevantes para a gestão de SLAs e SLOs.
- **Fornecer detalhes sobre a abertura de incidentes no ITSM**, permitindo uma análise completa do histórico de problemas e da performance da equipe de suporte.
- **Controlar SLAs e SLOs dos serviços fornecidos e da infraestrutura**, garantindo o cumprimento dos acordos de nível de serviço e a satisfação dos clientes.

Adicionalmente, o modelo de dados facilitará a análise centralizada de informações de monitoração, inventário e ITSM, permitindo a identificação de tendências, gargalos e anomalias que possam impactar os serviços. A integração dos dados visa agilizar a resolução de problemas e fornecer insights para a tomada de decisão estratégica, com base em informações completas e contextualizadas. Além disso, a aplicação Kali oferecerá um portal interativo e dashboards dinâmicos, permitindo que os usuários acompanhem em tempo real a saúde da infraestrutura e realizem ações gerenciais baseadas nos dados.

### 3. Situação Problemática:

Atualmente, a empresa enfrenta desafios na gestão de sua infraestrutura de TI e na garantia da qualidade dos serviços, devido à falta de uma solução centralizada para armazenar e analisar os dados gerados por suas diversas ferramentas de monitoração. Essa falta de integração resulta em:

- **Dados em silos:** As informações ficam dispersas em diferentes sistemas, dificultando a obtenção de uma visão unificada do ambiente e a correlação.
- **Dificuldade na análise de dados:** A análise de informações relevantes para a tomada de decisão torna-se complexa e trabalhosa, demandando tempo e recursos.
- **Gargalos na resolução de problemas:** A identificação da causa raiz de problemas e a tomada de ações corretivas.
- **Gestão de SLAs e SLOs:** A falta de controle e visibilidade sobre a disponibilidade dos serviços e a performance da infraestrutura aumenta o risco de não cumprimento dos acordos de nível de serviço.

### 4. Modelos de negócio:

O modelo de dados desenvolvido é focado na integração entre monitoração, inventário e gestão de incidentes ITSM. O banco permitirá: Registro centralizado de alertas e incidentes, monitoramento em tempo real da infraestrutura, Correlação entre falhas, ativos e SLAs, melhorando a gestão operacional e um Portal interativo para usuários, permitindo a visualização e gerenciamento dos alertas e incidentes em um ambiente acessível via dashboard.



## 5. Diagrama da base dados:

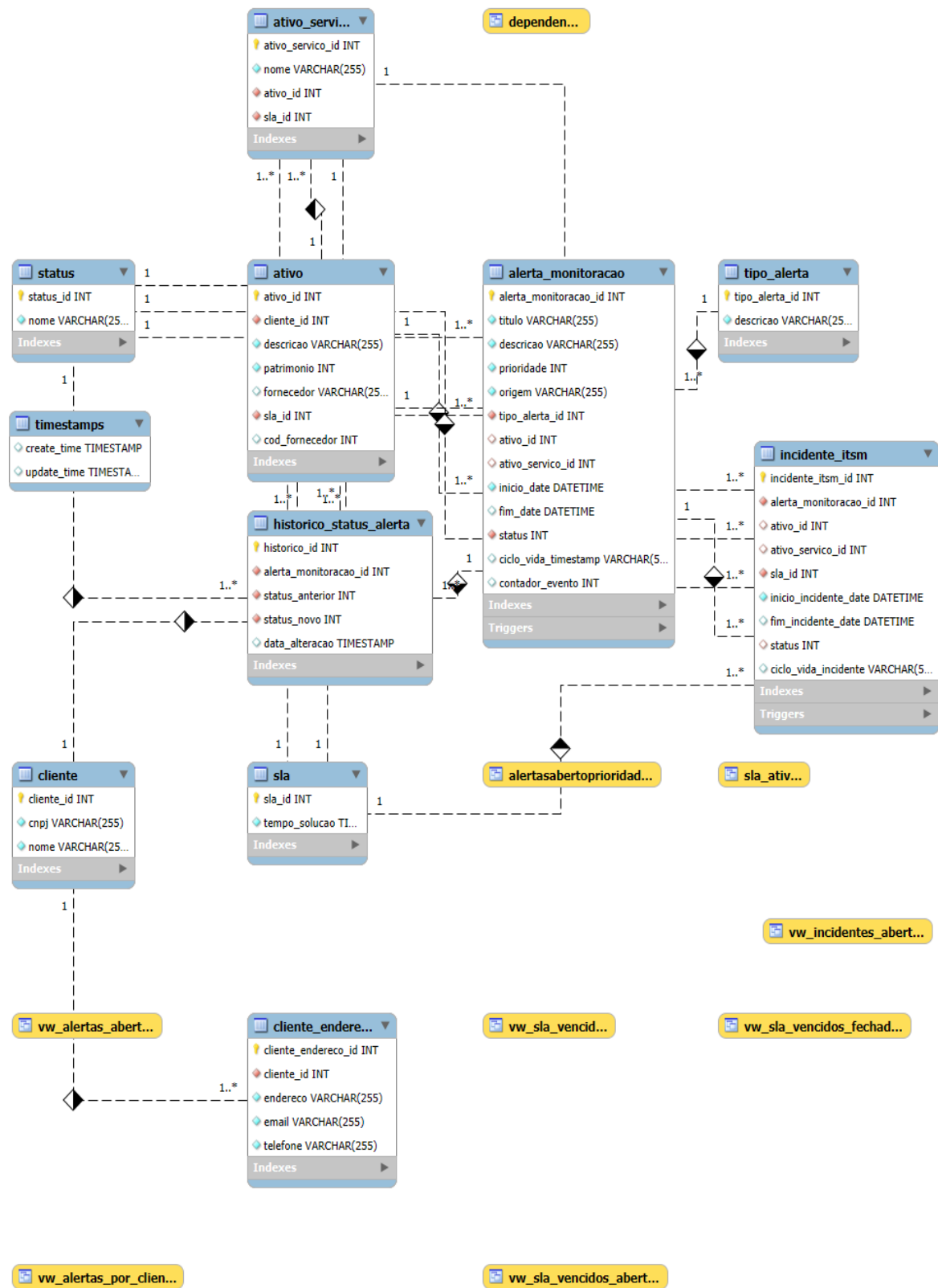
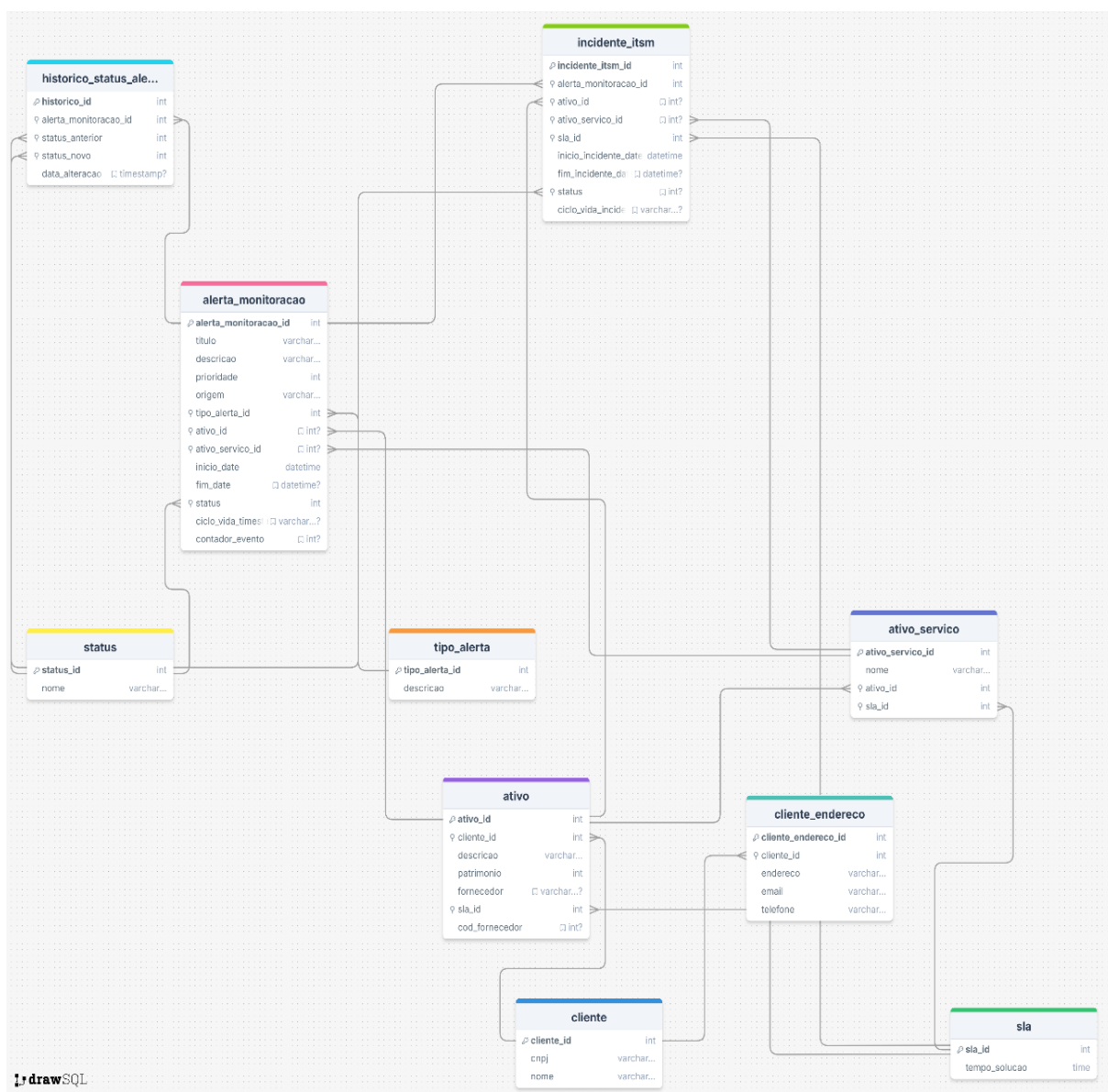


Tabela	Descrição
Cliente	Registra os clientes do sistema.
cliente_endereco	Endereços dos clientes.
Sla	Regras de Acordo de Nível de Serviço.
Ativo	Equipamentos e dispositivos monitorados.
ativo_servico	Serviços associados a ativos.
tipo_alerta	Classificação de alertas de monitoração.
alerta_monitoracao	Registros de alertas ativos.
incidente_itsm	Registros de incidentes associados a alertas.
historico_status_alerta	Armazena histórico de mudanças no status dos alertas.
Status	Status possíveis para alertas e incidentes.



## 6. Principais Relacionamentos

- **cliente → cliente\_endereco (1:N)** : Um cliente pode ter vários endereços cadastrados. Chave estrangeira: cliente\_endereco.cliente\_id → cliente.cliente\_id.
- **cliente → sla (1:N)** : Um cliente pode ter vários SLAs associados, mas um SLA pertence a apenas um cliente. Chave estrangeira: sla.cliente\_id → cliente.cliente\_id.
- **ativo → ativo\_servico (1:N)** : Um ativo pode fornecer vários serviços. Chave estrangeira: ativo\_servico.ativo\_id → ativo.ativo\_id.
- **ativo → alerta\_monitoracao (1:N) 1** : Um ativo pode gerar vários alertas de monitoração ao longo do tempo. Chave estrangeira: alerta\_monitoracao.ativo\_id → ativo.ativo\_id.
- **tipo\_alerta → alerta\_monitoracao (1:N)** : Um tipo de alerta pode estar associado a múltiplos alertas registrados. Chave estrangeira: alerta\_monitoracao.tipo\_alerta\_id → tipo\_alerta.tipo\_alerta\_id.
- **alerta\_monitoracao → incidente\_itsm (1:1)** : Um alerta pode originar um único incidente no ITSM, e cada incidente ITSM está vinculado a apenas um alerta. Chave estrangeira: incidente\_itsm.alerta\_monitoracao\_id → alerta\_monitoracao.alerta\_monitoracao\_id.
- **sla → incidente\_itsm (1:N)** : Um SLA pode estar associado a vários incidentes ITSM, garantindo controle sobre tempos de resposta e resolução. Chave estrangeira: incidente\_itsm.sla\_id → sla.sla\_id.
- **incidente\_itsm → historico\_status\_alerta (1:N)** : Um incidente pode ter múltiplos registros de alteração de status, criando um histórico completo. Chave estrangeira: historico\_status\_alerta.incidente\_itsm\_id → incidente\_itsm.incidente\_itsm\_id.
- **status → historico\_status\_alerta e incidente\_itsm (1:N)** : A tabela status contém os possíveis estados para alertas e incidentes.
  - Chave estrangeira: historico\_status\_alerta.status\_id → status.status\_id.
  - Chave estrangeira: incidente\_itsm.status\_id → status.status\_id.
- **ativo\_servico → sla (1:N)** : Um SLA pode ser associado a vários serviços fornecidos por ativos, garantindo controle sobre disponibilidade e desempenho. Chave estrangeira: sla.ativo\_servico\_id → ativo\_servico.ativo\_servico\_id.

## 7. Views

O banco de dados inclui diversas Views para facilitar a extração e análise de dados.:

```
SHOW FULL TABLES WHERE Table_type = 'VIEW';
```

```
1 • SHOW FULL TABLES WHERE Table_type = 'VIEW';
```

Tables_in_kali	Table_type
alertasabertprioridades	VIEW
dependencia	VIEW
sla_ativos	VIEW
vw_alertas_abertos	VIEW
vw_alertas_por_cliente	VIEW
vw_incidentes_abertos	VIEW
vw_sla_vencidos	VIEW
vw_sla_vencidos_abertos	VIEW
vw_sla_vencidos_fechados	VIEW

- **alertasabertprioridades** : Todos os alertas que estão em abertos e sua prioridade.

query-validação incidente\_itsm incidente\_itsm alerta\_monitoracao status incidente\_itsm SQL File 8\* alertasabertpri

Limit to 1000 rows

```
1 • SELECT * FROM kali.alertasabertprioridades;
```

alerta_monitoracao_id	titulo	alerta_descricao	prioridade	alerta_status	aler
86	Servidor Linux - Uso Alto de CPU	CPU a 95% devido a processos de log	3	1	2025
65	Roteador Mikrotik RB3011 - Memória Alta	Uso de memória RAM atingiu 90%	3	1	2025
9	Erro de Roteamento HP Aruba 2930F	Possível configuração incorreta no routing VLAN...	3	1	2025
63	Roteador Mikrotik RB3011 - CPU Elevada	Processos de Firewall consumindo 95% da CPU	3	1	2025
53	Switch Cisco Catalyst 9200 - CPU Elevada	Uso de CPU em 85% devido a alto tráfego	3	1	2025
38	CPU Elevada no Firewall Palo Alto	Inspeção SSL causando alta carga na CPU	3	1	2025
13	Alto Uso de CPU no Mikrotik RB3011	Processo de Firewall/Queue consumindo 90% d...	3	1	2025
57	Switch Dell PowerSwitch N2024 - Alta Memória	Uso de memória superior a 90%	3	1	2025
80	Roteador Mikrotik RB3011 - Memória Alta	Uso de memória RAM atingiu 90%	3	1	2025
78	Roteador Mikrotik RB3011 - CPU Elevada	Processos de Firewall consumindo 95% da CPU	3	1	2025
73	Switch Cisco Catalyst 9200 - CPU Elevada	Uso de CPU em 85% devido a alto tráfego	3	1	2025
60	Switch HP Aruba 2930F - QoS Violado	Excesso de tráfego não prioritário ultrapassand...	2	1	2025
70	Roteador Cisco ISR 4321 - VPN Offline	Usuários sem acesso remoto	2	1	2025

- **dependência** : Mostrar os serviços suas dependências:

Limit to 1000 rows

```
1 • SELECT * FROM kali.dependencia;
```

ativo_servico_id	nome_servico	ativo_id	descricao_ativo
1	Link Dedicado 100Mbps	1	Switch Cisco Catalyst 9200
2	Fibra Óptica 1Gbps	2	Switch HP Aruba 2930F
3	Internet Corporativa 500Mbps	3	Roteador Mikrotik RB3011
4	Backup de Link 200Mbps	4	Roteador Cisco ISR 4321
5	VPN MPLS	5	Access Point Ubiquiti UniFi UAP-AC-PRO
6	Firewall Gerenciado	6	Firewall Palo Alto PA-820
7	Antivírus Corporativo	7	Firewall Fortinet FortiGate 100E
8	Monitoramento de Ameaças	8	Switch Dell PowerSwitch N2024
9	Gestão de Logs (SIEM)	9	Switch Juniper EX2300
10	Proteção DDoS	10	Roteador Huawei AR 1220
11	Telefonia IP (PABX Virtual)	11	Firewall Check Point 6200
12	Número 0800 Empresarial	12	Firewall SonicWall NSA 2700
13	Ramal SIP para Home Office	13	Firewall Sophos XG 210
14	Central de Atendimento Omni...	14	Firewall WatchGuard Firebox T80

- **sla\_ativos** : lista todos ativos e SLA para cada componente.

query-validação alerta\_monitoracao **sla\_ativos**

Limit to 1000 rows

```
1 SELECT * FROM sla_ativos;
```

id	ativo_servico_id	ativo_descricao	sla_id	tempo_solucao	alerta_monitoracao_id	alerta_titulo	alerta_descricao
5	NULL	Access Point Ubiquiti UniFi...	5	05:00:00	81	Firewall Palo Alto - Ataques Detectados	IPS bloqueou tentativa de exploração via SSH
7	NULL	Firewall Fortinet FortGate...	7	07:00:00	83	Access Point Ubiquiti - Queda de Conexão	AP não responde a pings na rede wireless
3	NULL	Roteador MikroTik RB3011	3	03:00:00	85	Roteador MikroTik RB3011 - Queda de Link S...	Link backup caiu, operação na última rota dispo...
9	NULL	Switch Juniper EX2300	9	09:00:00	86	Servidor Linux - Uso Alto de CPU	CPU a 95% devido a processos de log
11	NULL	Firewall Check Point 6200	1	01:00:00	88	Servidor Windows - Serviços Parados	Vários serviços não iniciaram corretamente após...
1	NULL	Switch Cisco Catalyst 9200	1	01:00:00	89	Teste Alerta	Descrição do alerta
5	NULL	Access Point Ubiquiti UniFi...	5	05:00:00	90	Título do Alerta	Descrição do Alerta
NULL	1	Link Dedicado 100Mbps	1	01:00:00	1	Queda de Link no Switch Cisco Catalyst 9200	Monitor detectou ausência de resposta ICMP no...
NULL	1	Link Dedicado 100Mbps	1	01:00:00	2	Pacotes Perdidos no Switch Catalyst 9200	Taxa de perda de pacotes acima de 10% detec...
NULL	1	Link Dedicado 100Mbps	1	01:00:00	4	Erro de Roteamento no Switch Catalyst 9200	Rota de backup não ativa adequadamente, inv...

- **vw\_alertas\_abertos**: alertas que ainda estão sem solução:

Limit to 1000 rows

```
1 SELECT * FROM kali.vw_alertas_abertos;
```

id	titulo	descricao	prioridade	origem	tipo_alerta	status
1	Queda de Link no Switch Cisco Catalyst 9200	Monitor detectou ausência de resposta ICMP no...	2	Monitoramento	Queda de Link Principal	Sucesso
2	Pacotes Perdidos no Switch Catalyst 9200	Taxa de perda de pacotes acima de 10% detec...	1	Monitoramento	Latência Elevada	Sucesso
3	Erro de Roteamento no Switch Catalyst 9200	Rota de backup não ativa adequadamente, inv...	2	Monitoramento	Erro de Roteamento	Sucesso
4	Memória Alta no Switch Catalyst 9200	Uso de memória acima de 90%, risco de travam...	2	Monitoramento	Servidor com Alto Uso de Memória	Sucesso
5	Queda de Link no Switch HP Aruba 2930F	Sem resposta no uplink VLAN trunk. Suspeita de...	2	Monitoramento	Queda de Link Principal	Sucesso
6	Perda de Pacotes no Switch HP Aruba 2930F	Monitor registrou 15% de perda de pacotes em ...	1	Monitoramento	Latência Elevada	Sucesso
7	Uso de CPU Alto no Switch HP Aruba 2930F	CPU atinge 85% devido a processo STP e broad...	2	Monitoramento	Servidor com Alto Uso de CPU	Sucesso
8	Erro de Roteamento HP Aruba 2930F	Possível configuração incorreta no routing VLAN...	3	Monitoramento	Erro de Roteamento	Sucesso
9	Queda de Link no Roteador MikroTik RB3011	Perda total de conectividade WAN. Provedor fo...	1	Monitoramento	Queda de Link Principal	Ruim
10	Alto Uso de CPU no MikroTik RB3011	Processo de Firewall/Queue consumindo 90% d...	3	Monitoramento	Servidor com Alto Uso de CPU	Ruim
11	Alto Uso de Memória no MikroTik RB3011	Memória acima de 85%: logs e cache DNS ocup...	1	Monitoramento	Servidor com Alto Uso de Memória	Ruim
12	Perda de Pacotes no MikroTik RB3011	Usuários reclamam de lentidão; monitor registro...	2	Monitoramento	Latência Elevada	Ruim
13	Queda de Link no Firewall Palo Alto PA-820	Interface externa down, sem acesso à internet	1	Firewall	Queda de Link Principal	Fuor de Serviço

- **vw\_incidentes\_por\_cliente** : contador os incidentes ITSM associados por cliente:

incidente\_itsm incidente\_itsm alerta\_monitoracao status

Limit to 1000 rows

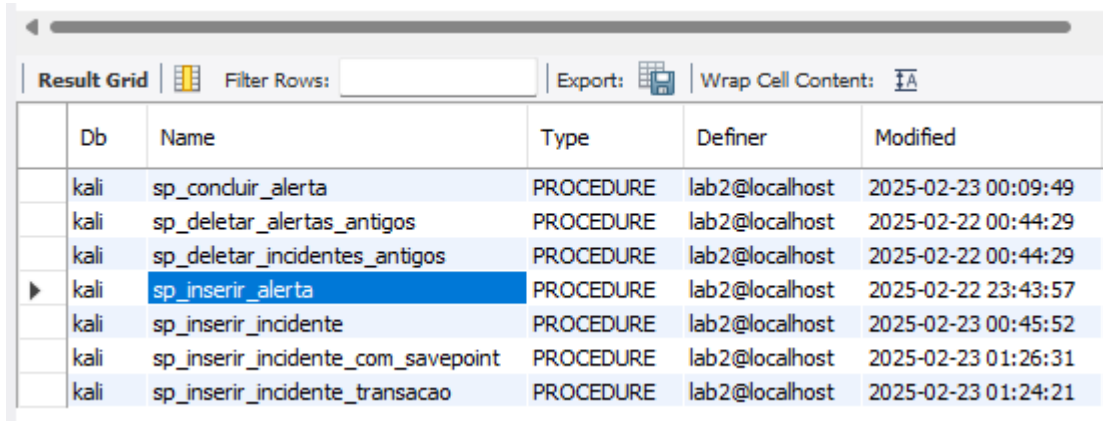
```
1 SELECT * FROM kali.vw_alertas_por_cliente;
```

cliente	total_alertas
Empresa Alpha Ltda	16
Beta Tech Solutions	15
Gamma Telecom	11
Zeta Consultoria	6
Delta Infraestrutura	7
Epsilon Engenharia	6
Eta Logística	1
Theta Segurança Digital	1
Iota Healthcare	1
Kappa Finance	1
Lambda Softwares	1



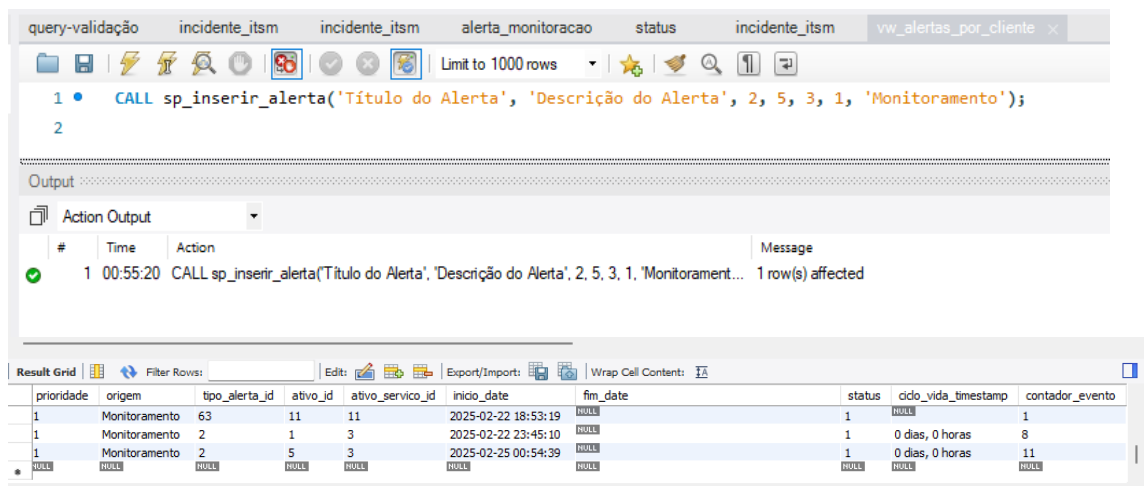
## 8. Store Procedures

O banco de dados inclui diversas Stored Procedures para automatizar processos. Abaixo estão as principais e como validá-las:



	Db	Name	Type	Definer	Modified
	kali	sp_conduir_alerta	PROCEDURE	lab2@localhost	2025-02-23 00:09:49
	kali	sp_deletar_alertas_antigos	PROCEDURE	lab2@localhost	2025-02-22 00:44:29
	kali	sp_deletar_incidentes_antigos	PROCEDURE	lab2@localhost	2025-02-22 00:44:29
▶	kali	sp_inserir_alerta	PROCEDURE	lab2@localhost	2025-02-22 23:43:57
	kali	sp_inserir_incidente	PROCEDURE	lab2@localhost	2025-02-23 00:45:52
	kali	sp_inserir_incidente_com_savepoint	PROCEDURE	lab2@localhost	2025-02-23 01:26:31
	kali	sp_inserir_incidente_transacao	PROCEDURE	lab2@localhost	2025-02-23 01:24:21

- **sp\_inserir\_alerta** : Verifica se um alerta já existe no banco, se não existir, insere um novo alerta. Se já existir, atualiza o contador de reincidência.



```
1 • CALL sp_inserir_alerta('Título do Alerta', 'Descrição do Alerta', 2, 5, 3, 1, 'Monitoramento');
2
```

Output

Action Output

#	Time	Action	Message
1	00:55:20	CALL sp_inserir_alerta('Título do Alerta', 'Descrição do Alerta', 2, 5, 3, 1, 'Monitorament...	1 row(s) affected

Result Grid

	prioridade	origem	tipo_alerta_id	ativo_id	ativo_servico_id	inicio_date	fin_date	status	ciclo_vida_timestamp	contador_evento
1	1	Monitoramento	63	11	11	2025-02-22 18:53:19	NULL	1	NULL	1
1	1	Monitoramento	2	1	3	2025-02-22 23:45:10	NULL	1	0 dias, 0 horas	8
1	1	Monitoramento	2	5	3	2025-02-25 00:54:39	NULL	1	0 dias, 0 horas	11
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- **sp\_deletar\_alertas\_antigos** // **sp\_deletar\_incidentes\_antigos** : Remove alertas antigos que não possuem mais relevância. 90 dias com status = 7 fechado

```
CREATE DEFINER=`lab2`@`localhost` PROCEDURE
`sp_deletar_alertas_antigos`()
BEGIN
    DELETE FROM alerta_monitoracao
    WHERE status = 7
    AND inicio_date < DATE_SUB(NOW(), INTERVAL 60 DAY);
END
```

- **sp\_concluir\_alerta:** Atualiza o status do alerta para "Fechado" e registra a data de conclusão.

```
SQL File 11*  incidente_itsm  alerta_monitoracao x
1 CALL sp_concluir_alerta(3, 1, 1, NOW());
2
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [iA](#)

mensagem\_sucesso

Alerta ID 1 concluído com sucesso.

- **sp\_alterar\_status\_incidente:** Modifica o status de um incidente conforme sua evolução.

```
SQL File 11*  incidente_itsm  alerta_monitoracao  sp_deletar_alertas_antigos - Ro...  sp_deletar_incidentes_antigos - ...  sp_alterar_status_incidente - Ro...
1 SELECT * FROM kali.incidente_itsm;
2 CALL sp_alterar_status_incidente(2, 6);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: [iA](#)

	incidente_itsm_id	alerta_monitoracao_id	ativo_id	ativo_servico_id	sla_id	inicio_incidente_date	fin_incidente_date	status	ciclo_vida_incidente
1	1	1	1	1	1	2025-02-12 18:19:50	2025-02-26 00:14:46	6	13 dias, 5 horas
2	2	1	1	1	1	2025-02-13 18:19:50	2025-02-26 00:20:14	6	12 dias, 6 horas
3	3	1	1	1	1	2025-02-14 18:19:50		1	

- **sp\_concluir\_alerta :** Marca um alerta como concluído, alterando seu status para 7 (Fechado)

SQL File 11\* incidente\_itsm alerta\_monitoracao x incidente\_itsm

```
1 CALL sp_concluir_alerta(3, 1, 1, NOW());
2
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [iA](#)

mensagem\_sucesso

Alerta ID 1 concluído com sucesso.

- **sp\_inserir\_incidente\_com\_savepoint:** Insere um incidente utilizando SAVEPOINT, permitindo rollback parcial em caso de erro.

```
CREATE DEFINER=`lab2`@`localhost` PROCEDURE
`sp_inserir_incidente_com_savepoint`(
  IN p_incidente_itsm_id INT,
  IN p_alerta_monitoracao_id INT,
  IN p_ativo_id INT,
  IN p_ativo_servico_id INT,
  IN p_sla_id INT
)
BEGIN
  DECLARE EXIT HANDLER FOR SQLEXCEPTION
  BEGIN
    ROLLBACK TO ponto_de_restore;
    SELECT 'ERRO: Transação revertida até o último SAVEPOINT.'
AS mensagem_erro;
  END;

  START TRANSACTION;

  SAVEPOINT ponto_de_restore;

  -- Inserção do incidente
  INSERT INTO incidente_itsm (
    incidente_itsm_id, alerta_monitoracao_id, ativo_id,
    ativo_servico_id, sla_id, inicio_incidente_date, status
  ) VALUES (
    p_incidente_itsm_id, p_alerta_monitoracao_id, p_ativo_id,
    p_ativo_servico_id, p_sla_id, NOW(), 1
  );

  -- Se der erro depois do SAVEPOINT, podemos reverter apenas a
última parte
  SAVEPOINT ponto_final;

  -- Simulação de erro (remova esse comentário para testar
rollback parcial)
  -- SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Erro simulado
para testar SAVEPOINT';

  COMMIT;
  SELECT 'Incidente inserido com sucesso!' AS mensagem_sucesso;
END
```

- **sp\_inserir\_incidente\_transacao** : Insere um incidente utilizando transações (START TRANSACTION, COMMIT, ROLLBACK).

```
CREATE DEFINER=`lab2`@`localhost` PROCEDURE CREATE
DEFINER=`lab2`@`localhost` PROCEDURE
`sp_inserir_incidente_transacao`(
    IN p_incidente_itsm_id INT,
    IN p_alerta_monitoracao_id INT,
    IN p_ativo_id INT,
    IN p_ativo_servico_id INT,
    IN p_sla_id INT
)
BEGIN
    DECLARE EXIT HANDLER FOR SQLEXCEPTION
    BEGIN
        ROLLBACK;
        SELECT 'ERRO: Transação revertida devido a um erro.' AS
mensagem_erro;
    END;
    START TRANSACTION;
    -- Verifica se o ID já existe
    IF (SELECT COUNT(*) FROM incidente_itsm WHERE incidente_itsm_id
= p_incidente_itsm_id) > 0 THEN
        ROLLBACK;
        SELECT 'ERRO: Incidente já existe!' AS mensagem_erro;
    ELSE
        INSERT INTO incidente_itsm (
            incidente_itsm_id, alerta_monitoracao_id, ativo_id,
ativo_servico_id, sla_id, inicio_incidente_date, status
        ) VALUES (
            p_incidente_itsm_id, p_alerta_monitoracao_id,
p_ativo_id, p_ativo_servico_id, p_sla_id, NOW(), 1
        );
        COMMIT;
        SELECT CONCAT('Incidente ID ', p_incidente_itsm_id, '
inserido com sucesso.') AS mensagem_sucesso;
    END IF;
END
```

## 9. Triggers

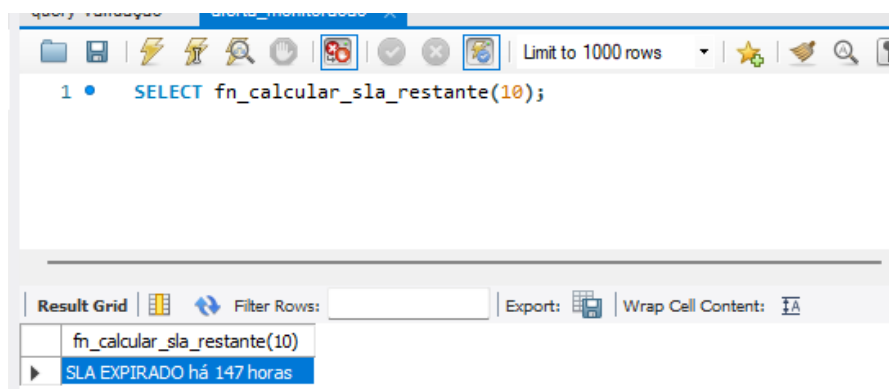
O banco de dados inclui diversas Triggers para garantir a integridade dos dados e automatizar processos.:

- **trg\_registrar\_mudanca\_status** : Registra automaticamente mudanças no status de alertas na tabela historico\_status\_alerta.
- **trg\_atualizar\_ciclo\_vida\_alerta** : Atualiza o campo ciclo\_vida no momento do fechamento do alerta.
- **trg\_atualizar\_ciclo\_vida\_incidente** : Atualiza o campo ciclo\_vida\_incidente ao encerrar um incidente.
- **trg\_auto\_inserir\_historico\_status** : Insere automaticamente um novo registro na tabela historico\_status\_alerta sempre que um alerta tem seu status alterado.

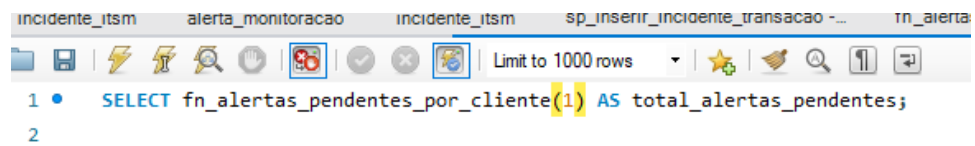
## 10. Funções

O banco de dados inclui diversas Funções para facilitar a manipulação e consulta de dados. Abaixo estão as principais Funções e como validá-las:

- **fn\_tempo\_medio\_resolucao**: Retorna o tempo restante para a conclusão do SLA de um incidente.

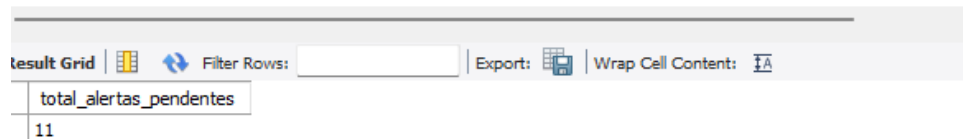


- **fn\_calcular\_sla\_restante**: Verifica se um incidente já foi aberto para um alerta específico.
- **fn\_contar\_alertas\_por\_ativo** : Retorna o número total de alertas associados a um ativo.
- **fn\_tempo\_medio\_resolucao** : Retorna o tempo médio de resolução dos incidentes fechados.
- **fn\_alertas\_pendentes\_por\_cliente** : Retorna soma de alerta por cliente.



The screenshot shows a SQL query editor with a toolbar at the top. The toolbar includes icons for file operations (save, open, print), editing (undo, redo, copy, paste), and execution (run, stop). A dropdown menu is set to "Limit to 1000 rows". The query text is as follows:

```
1 • SELECT fn_alertas_pendentes_por_cliente(1) AS total_alertas_pendentes;  
2
```

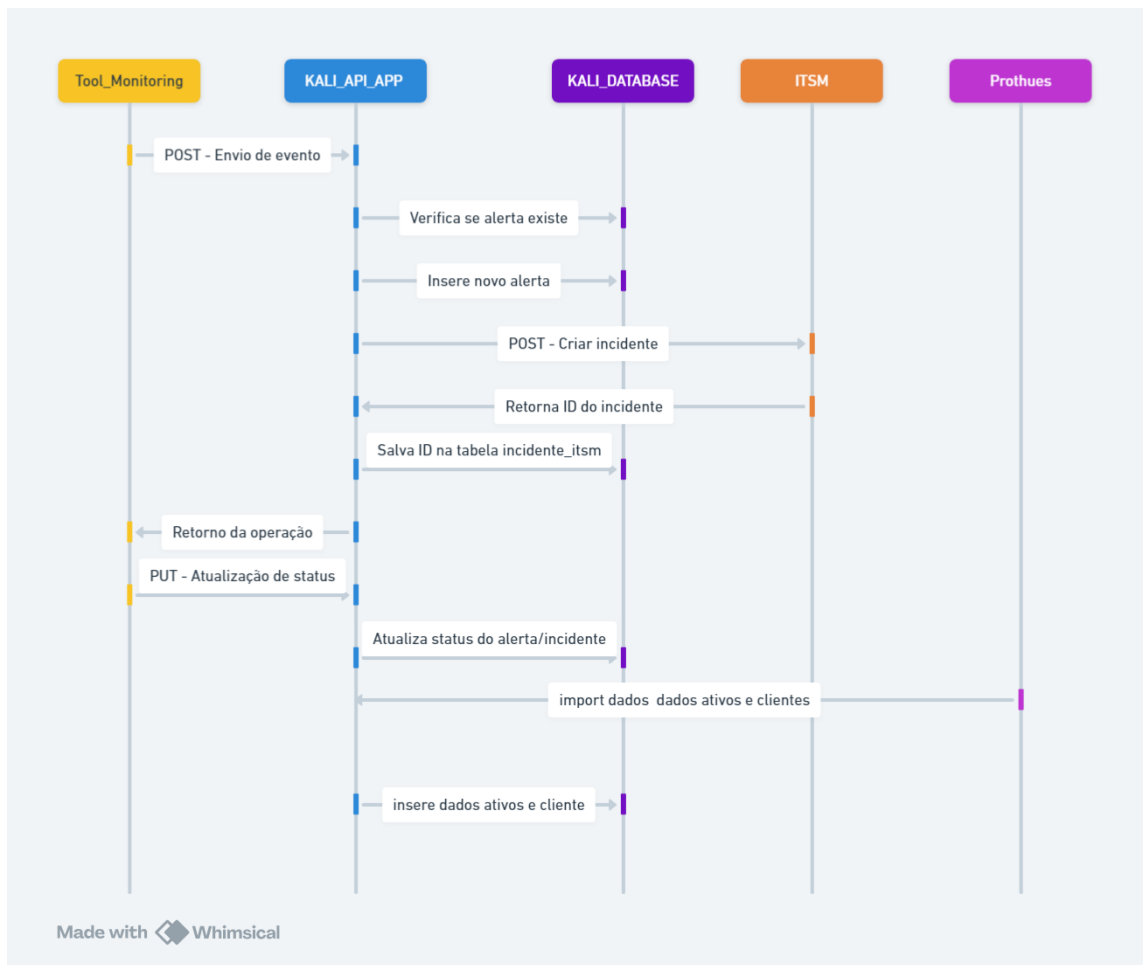


The screenshot shows a "Result Grid" with a toolbar at the top. The toolbar includes icons for grid view, filter rows, export, and wrap cell content. The grid contains one row with the following data:

total_alertas_pendentes
11

## 11. Inserção de dados pela Aplicação

A inserção de dados no banco de dados será realizada por meio da aplicação Kali, desenvolvida em Python e integrada com um API Gateway. Esse gateway possibilita a comunicação segura entre as ferramentas de monitoração, a aplicação e o banco de dados.



### Fluxo de Inserção de Dados

1. As ferramentas de monitoração enviam eventos via POST para a aplicação.
2. A aplicação valida se o alerta já existe na base de dados:
  - Se o alerta já existir, ele é atualizado.
  - Se o alerta não existir, ele é inserido na tabela alerta\_monitoracao.
3. Se for um novo alerta, a aplicação:
  - Realiza um POST no ITSM para abertura de um incidente.
  - O ID do incidente retornado pelo ITSM é utilizado como chave primária na tabela incidente\_itsm.
4. Alterações no status dos alertas ou incidentes são feitas via chamadas PUT.
5. Os usuários podem visualizar, gerenciar e interagir com os alertas e incidentes através de um dashboard interativo.

### Chamadas de API e Procedimentos Envolvidos

A aplicação chama as Stored Procedures para garantir que a inserção e atualização:

- **sp\_inserir\_alerta:** Verifica a existência do alerta e o insere ou atualiza se necessário.
- **sp\_abrir\_incidente\_para\_alerta:** Cria um incidente ITSM vinculado a um alerta.
- **sp\_concluir\_alerta:** Fecha um alerta quando resolvido.
- **sp\_alterar\_status\_incidente:** Modifica o status do incidente ITSM conforme sua evolução.

Esse fluxo garante que os dados de monitoramento sejam registrados, processados e atualizados.

### 12. Análise e Relatórios analíticos

A extração e análise de informações do banco de dados foram realizadas utilizando ferramentas especializadas para fornecer insights estratégicos sobre a infraestrutura monitorada: METABASE que permite conexão direto do banco e cria dashboard e relatórios.





### 13. Backup e Alta Disponibilidade

Rotina de Backup para garantir a integridade e recuperação dos dados, foi implementado um processo automatizado de backup:

- Backup Diário: Um dump do banco de dados é gerado automaticamente todas as noites.
- Retenção: Backups são armazenados por um período de 30 dias.
- Armazenamento Seguro: Os backups são transferidos para um servidor remoto seguro e armazenados em múltiplos locais para redundância.

```
mysqldump -u backup_user -p --routines --triggers --single-transaction kali > /backups/backup_kali_$(date +%Y-%m-%d).sql
```

Estratégia de Alta Disponibilidade e garantir resiliência, o banco de dados conta com:

- Replicação Master-Slave: O banco de dados primário replica seus dados para um servidor secundário em tempo real.
- Failover Automático: Caso o servidor principal falhe, a aplicação se conecta automaticamente ao servidor secundário.
- Balanceamento de Carga: Uso de ProxySQL ou HAProxy para distribuir as requisições entre os servidores disponíveis.

Essa abordagem garante que o banco de dados permaneça acessível e minimiza os impactos de falhas no sistema.

### 14. Permissões e usuários do banco

Usuário	Função	Nível de Permissão
admin_db	Administrador do Banco	ALL PRIVILEGES (acesso total ao banco kali)
backup_user	Usuário de Backup	SELECT, LOCK TABLES, SHOW VIEW, EVENT
app_user	Usuário da Aplicação	SELECT, INSERT, UPDATE, DELETE
analyst_user	Analista de Dados	SELECT, EXECUTE (pode rodar consultas e funções)
read_only	Usuário de Leitura	SELECT (apenas pode visualizar os dados)

## 15. Conclusão

primeira etapa do Projeto Kali foi concluída com sucesso, incluindo a modelagem do banco de dados, a definição da estrutura de tabelas, a implementação de Stored Procedures, Triggers, Views e Funções. Essa fase permite uma base sólida para a gestão eficiente de alertas, incidentes e SLAs, garantindo que as informações sejam registradas, validadas e processadas de forma otimizada.

O próximo passo do projeto está diretamente relacionado à trilha de desenvolvimento pessoal sobre Back-end e Front-end, com objetivo de expandir as funcionalidades da aplicação Kali para permitir uma experiência completa e interativa aos usuários. Os próximos focos incluem: Back-end: Desenvolvimento da API REST, Implementação de autenticação e controle de acesso para usuários e melhorias na automação de processos, incluindo notificações e integrações externas. Front-end: Criação de um portal com dashboard interativo, Implementação de gráficos e relatórios dinâmicos.

### Links:

<https://drawsql.app/teams/edson/diagrams/kali-date>

[https://github.com/edsmont/kali\\_projeto\\_sql\\_coder](https://github.com/edsmont/kali_projeto_sql_coder)