Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Atelier 3 : Conception d'un Dashboard Observabilité Paiements avec Grafana

🧩 Objectifs pédagogiques

À la fin de cet atelier, le participant sera capable de :

1. Configurer une source de données InfluxDB dans Grafana pour collecter des métriques de paiements.
2. Créer un tableau de bord (Dashboard) d'observabilité e-Banking complet, structuré en plusieurs rangées (KPIs, tendances, comparaisons, analyses).
3. Élaborer des requêtes Flux pour extraire, agréger et visualiser les métriques clés (montant, taux de succès, latence, échecs).
4. Personnaliser les visualisations : Stat Panels, Graphiques, Tables, Bar Charts, avec couleurs, unités, et seuils.
5. Automatiser l'observabilité : configurer le rafraîchissement, les annotations et les seuils SLA (latence, taux de réussite).
6. Sauvegarder et partager un Dashboard professionnel prêt pour un environnement de production.

📋 Table des matières

1. Préparation & Environnement
2. Structure Globale du Dashboard
3. Panel 1 : Revenu Total (24h)
4. Panel 2 : Taux de Succès (%)
5. Panel 3 : Latence Moyenne
6. Panel 4 : Montant Échoué
7. Panel 5 : Tendance Volume (7j)
8. Panel 6 : Revenue par Devise
9. Panel 7 : Comparaison Jour/Jour
10. Panel 8 : Performance par Région
11. Panel 9 : Latence par Méthode

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

12. Panel 10 : Transactions par Méthode
13. Panel 11 : Performance Marques Cartes
14. Panel 12 : Santé Système

🔧 1. Préparation & Environnement

Étape 1.0 : Simuler les trafics de paiements

```
cd ~/Grafana-Stack/observability-stack
sudo ./simulate.sh
```

```
ubuntu@grafana-stack:~/Grafana-Stack/observability-stack$ sudo ./simulate.sh
Starting payment simulation...
API URL: http://localhost:8080/api/payments
Number of requests: 100
Max delay between requests: 2s
--------------------------------------
[1/100] SUCCESS: 950.00 EUR - Customer: cust_9721
[2/100] SUCCESS: 500.00 USD - Customer: cust_9830
[3/100] SUCCESS: 910.00 EUR - Customer: cust_3020
[4/100] SUCCESS: 590.00 EUR - Customer: cust_8420
[5/100] SUCCESS: 270.00 GBP - Customer: cust_2381
[6/100] SUCCESS: 780.00 GBP - Customer: cust_9925
[7/100] SUCCESS: 460.00 EUR - Customer: cust_8050
[8/100] SUCCESS: 730.00 EUR - Customer: cust_6061
[9/100] SUCCESS: 780.00 USD - Customer: cust_2591
[10/100] SUCCESS: 290.00 GBP - Customer: cust_5627
[11/100] SUCCESS: 10.00 EUR - Customer: cust_4577
[12/100] SUCCESS: 870.00 USD - Customer: cust_9227
[13/100] SUCCESS: 970.00 USD - Customer: cust_4094
[14/100] SUCCESS: 0.00 EUR - Customer: cust_5702
[15/100] SUCCESS: 320.00 USD - Customer: cust_1575
[16/100] SUCCESS: 10.00 GBP - Customer: cust_3432
[17/100] SUCCESS: 420.00 USD - Customer: cust_5683
[18/100] SUCCESS: 510.00 EUR - Customer: cust_2349
[19/100] SUCCESS: 480.00 EUR - Customer: cust_8151
[20/100] SUCCESS: 980.00 USD - Customer: cust_8950
[21/100] SUCCESS: 590.00 EUR - Customer: cust_6188
[22/100] SUCCESS: 20.00 EUR - Customer: cust_4024
[23/100] SUCCESS: 520.00 GBP - Customer: cust_4501
[24/100] SUCCESS: 210.00 GBP - Customer: cust_8677
[25/100] SUCCESS: 300.00 EUR - Customer: cust_3718
[26/100] SUCCESS: 120.00 GBP - Customer: cust_6556
[27/100] SUCCESS: 340.00 GBP - Customer: cust_9472
[28/100] SUCCESS: 710.00 USD - Customer: cust_5855
[29/100] SUCCESS: 270.00 USD - Customer: cust_5797
[30/100] SUCCESS: 900.00 GBP - Customer: cust_3307
[31/100] SUCCESS: 130.00 GBP - Customer: cust_7087
```

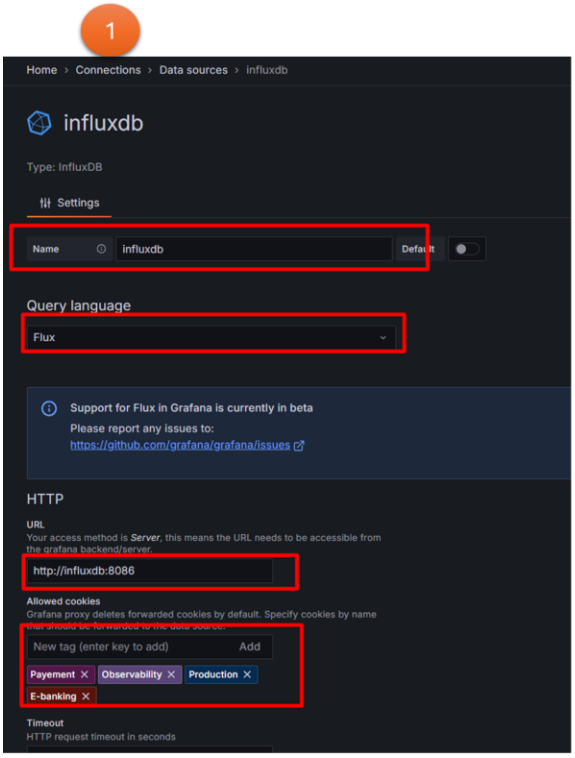
Étape 1.1 : Vérifier la Connexion InfluxDB

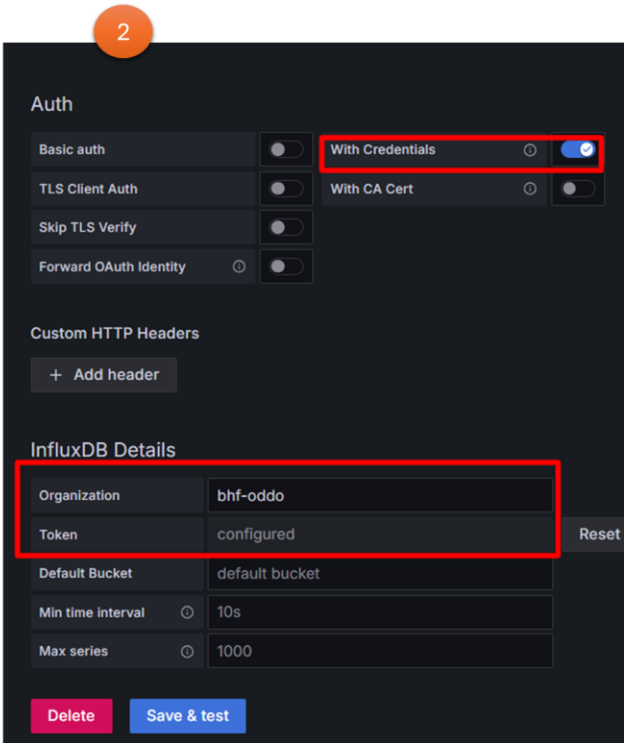
Grafana → Configuration → Data sources

1. Cliquez sur **"Add data source"**

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

2. Sélectionnez **InfluxDB**
3. **Configuration** :
    - **Name** : `InfluxDB-Payments`
    - **URL** : `http://influxdb:8086`
    - **Access** : `Server`
    - **InfluxDB Details** :
        - **Organization** : `bhf-oddo`
        - **Token** : Votre token API InfluxDB
        - **Default Bucket** : `payments`
    - **Query Language** : `Flux`
4. Cliquez **"Save & test"** ✅





Étape 1.2 : Créer le Dashboard

Grafana → + → Dashboard

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

1. Cliquez sur **"Create New" → "Dashboard"**
2. Panel vide s'affiche
3. Cliquez sur l'icône ⚙️ (Paramètres) en haut à droite

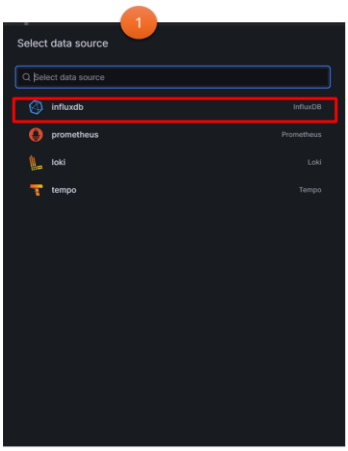
🧩 Étape 1.3 — Paramètres Généraux du Dashboard

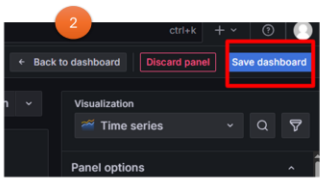
Section	Champ	Valeur / Action
General	**Title**	💳 ***Payment Analytics – Production Dashboard***
	Description	***Tableau de bord complet de supervision e-Banking, permettant de suivre en temps réel les volumes, taux de succès, tendances, et anomalies des transactions de paiement à partir de données InfluxDB.***
	Tags	**payments, financial, production, observability, sla, payments-system**
	Folder	***(Laisser vide ou choisir un dossier projet : "Payment Observability")***
	Editable	✅ ***Oui (coché)***
Time options	**Time zone**	***Browser Time***
	Auto refresh	**30s, 1m, 5m, 10m, 30m, 1h, 6h, 12h, 24h**
	Now delay	**0m**
Panel options	**Graph tooltip**	***Default***
	Preload panels	✅ ***Activé***

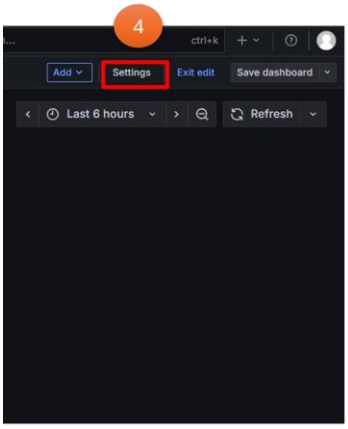
📝 Validation :

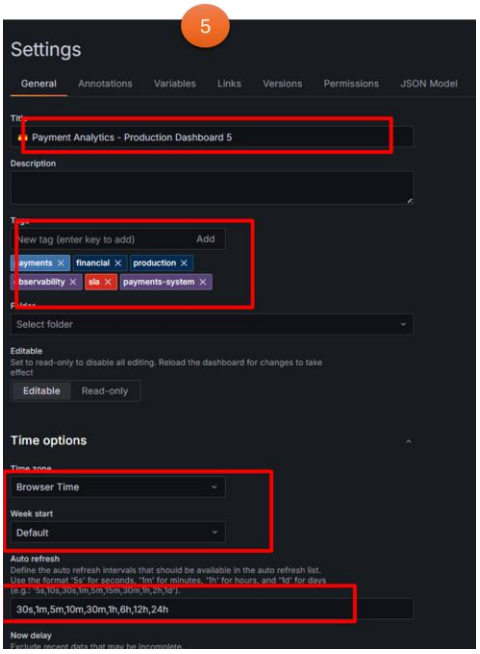
- **En haut à droite, cliquez sur 💾 Save Dashboard**
- **Le titre s'affiche : 💳 Payment Analytics – Production Dashboard**
- **Le tag principal payments apparaît dans la barre d'en-tête du tableau de bord.**

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com









Étape 1.3 : Ajouter Variables de Template

Settings → Variables → New

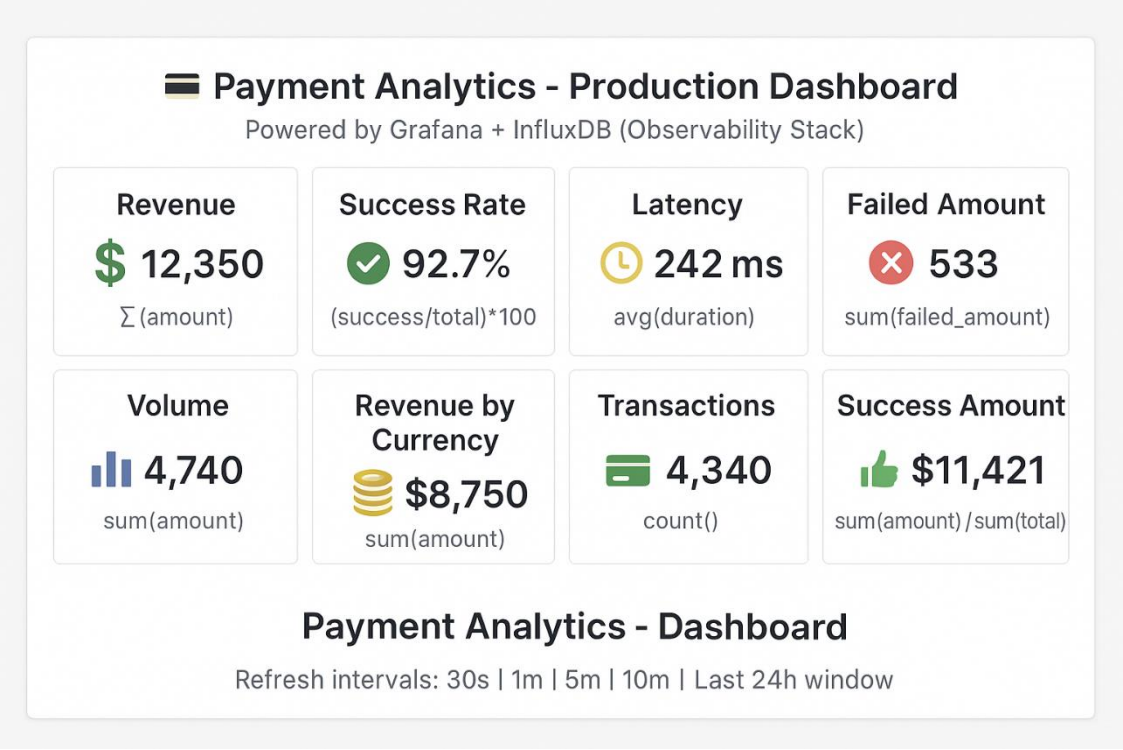
Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

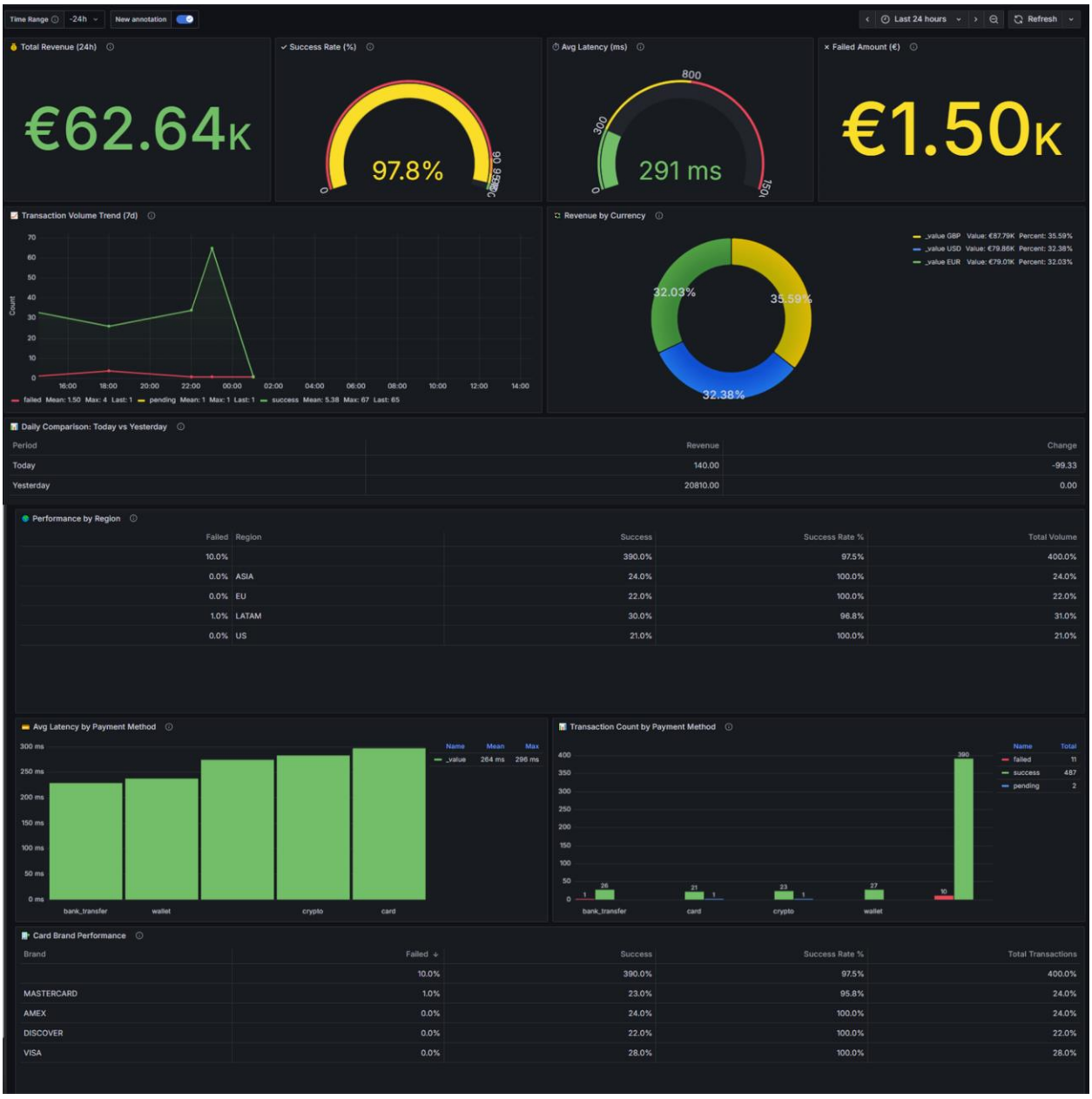
Variable 1 : Time Range

- **Name** : `timerange`
- **Type** : `Custom`
- **Custom Options** : `-24h,-7d,-30d,-90d`
- **Default** : `-24h`
- **Description** : `Time range for metrics analysis`

📐 2. Structure Globale du Dashboard



Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com



📊 Panel 1 : Revenu Total (24h)

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

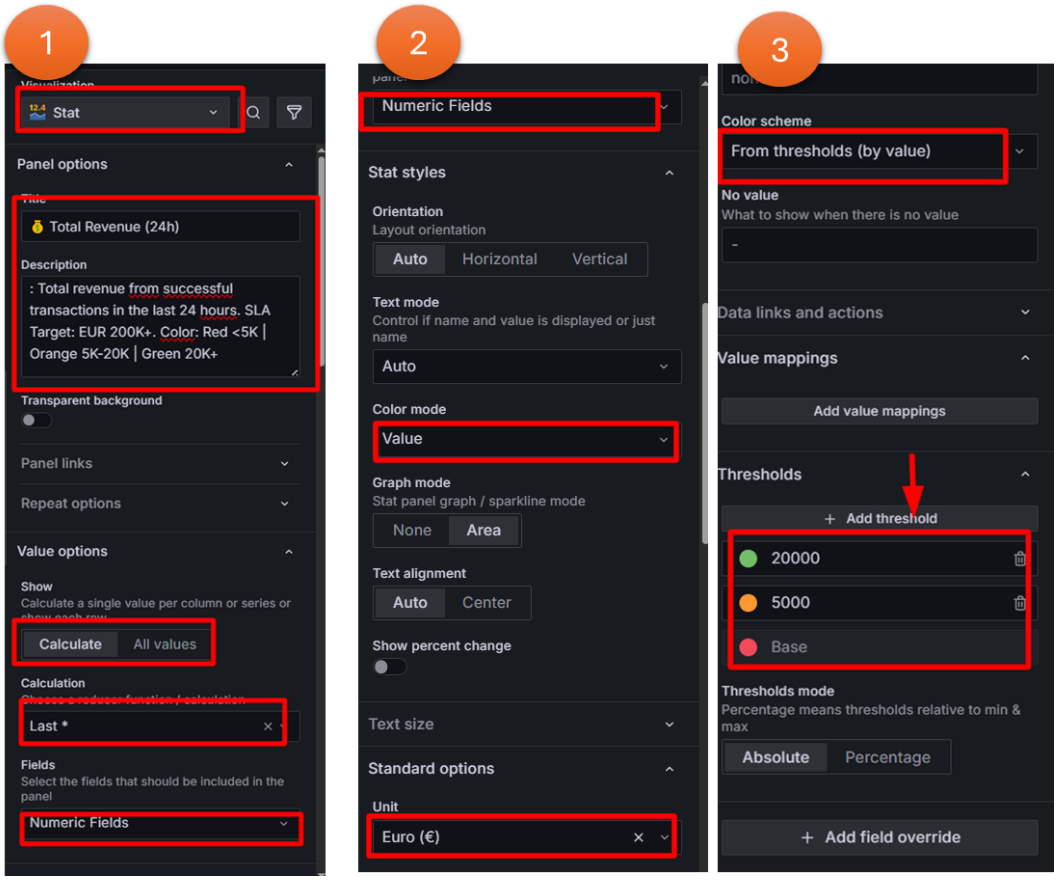
Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Type : Stat Card avec seuils colorés

Configuration GUI

Panel Settings

- **Panel Title** : 💰 `Total Revenue (24h)`
- **Panel Type** : `Stat`
- **Dimensions** : Position x:0, y:0 | Taille h:8, w:6
- **Description** : `Total revenue from successful transactions in the last 24 hours. SLA Target: EUR 200K+. Color: Red <5K | Orange 5K-20K | Green 20K+`



Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Field Configuration

Standard Options → Value Options

- **Unit** : `Currency (EUR)` → `currencyEUR`
- **Decimals** : `2`
- **Min** : (Vide)
- **Max** : (Vide)

Threshold Options

- **Threshold mode** : `Absolute`
- **Steps** :
  - Step 1 : `0` → `dark-red`
  - Step 2 : `5000` → `orange`
  - Step 3 : `20000` → `green`

Panel Options

Text Mode : `Auto` **Color Mode** : `Value` **Graph Mode** : `Area` **Justify Mode** : `Center` **Orientation** : `Auto`

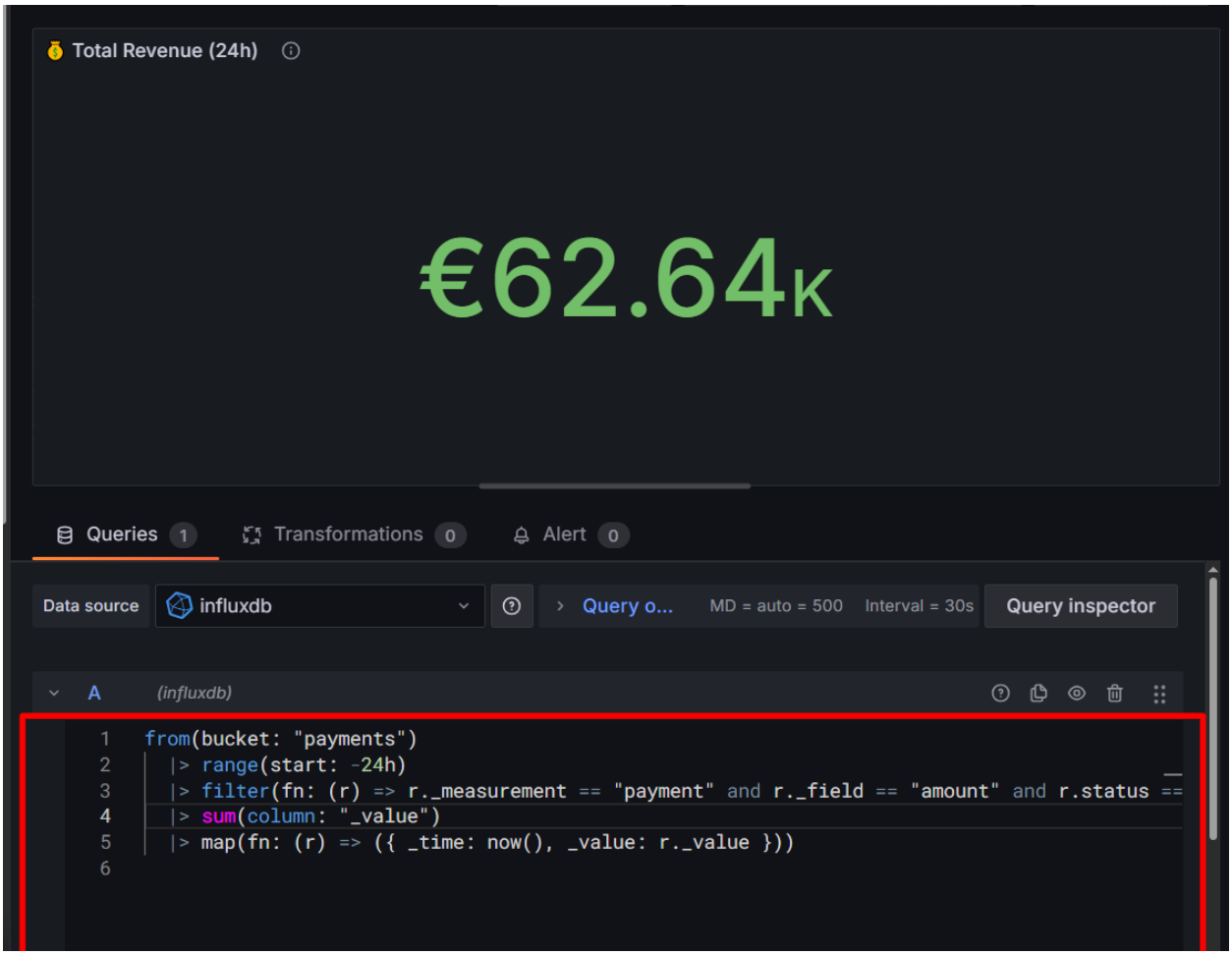
Requête InfluxDB (Flux)

```
from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) => r._measurement == "payment" and r._field == "amount" and r.status == "success")
  |> sum(column: "_value")
  |> map(fn: (r) => ({ _time: now(), _value: r._value }))
```

RefId : `A`

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

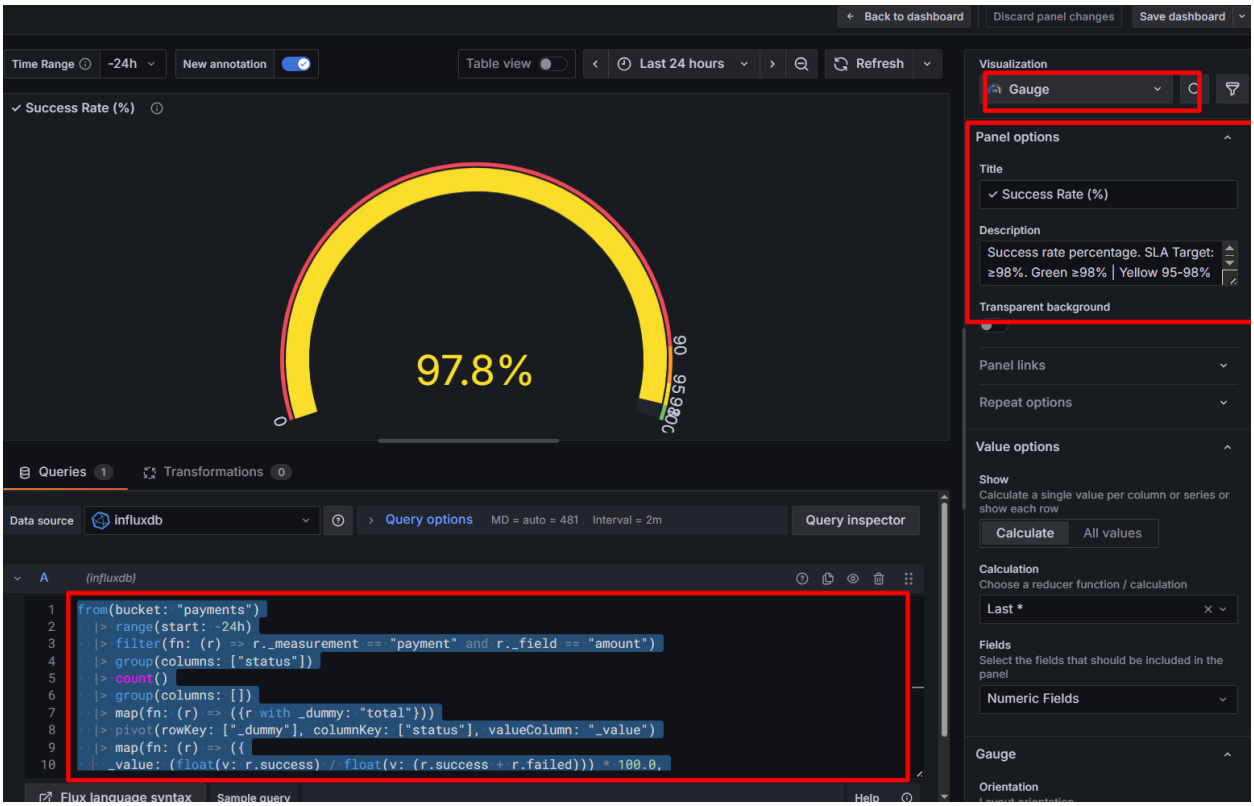


Cliquer Save as Dashbord

✅ Panel 2 : Taux de Succès (%)

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com



Type : Gauge avec arc

Configuration GUI

Panel Settings

- **Panel Title** : ✔ `Success Rate (%)`
- **Panel Type** : `Gauge`
- **Description** : `Success rate percentage. SLA Target: ≥98%. Green ≥98% | Yellow 95-98% | Red <95%. Real-time transaction reliability`

Field Configuration

Standard Options

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

- **Unit** : `Percent (0-100)`
- **Decimals** : `1`
- **Min** : `0`
- **Max** : `100`

Threshold Options

- **Mode** : `Absolute`
- **Steps** :
  - Step 1 : `0` → `red`
  - Step 2 : `90` → `orange`
  - Step 3 : `95` → `yellow`
  - Step 4 : `98` → `green`

Panel Options

Gauge Type : `Arc` **Show Threshold Labels** : `ON` **Show Threshold Markers** : `ON` **Orientation** : `Auto`

Requête InfluxDB (Flux)

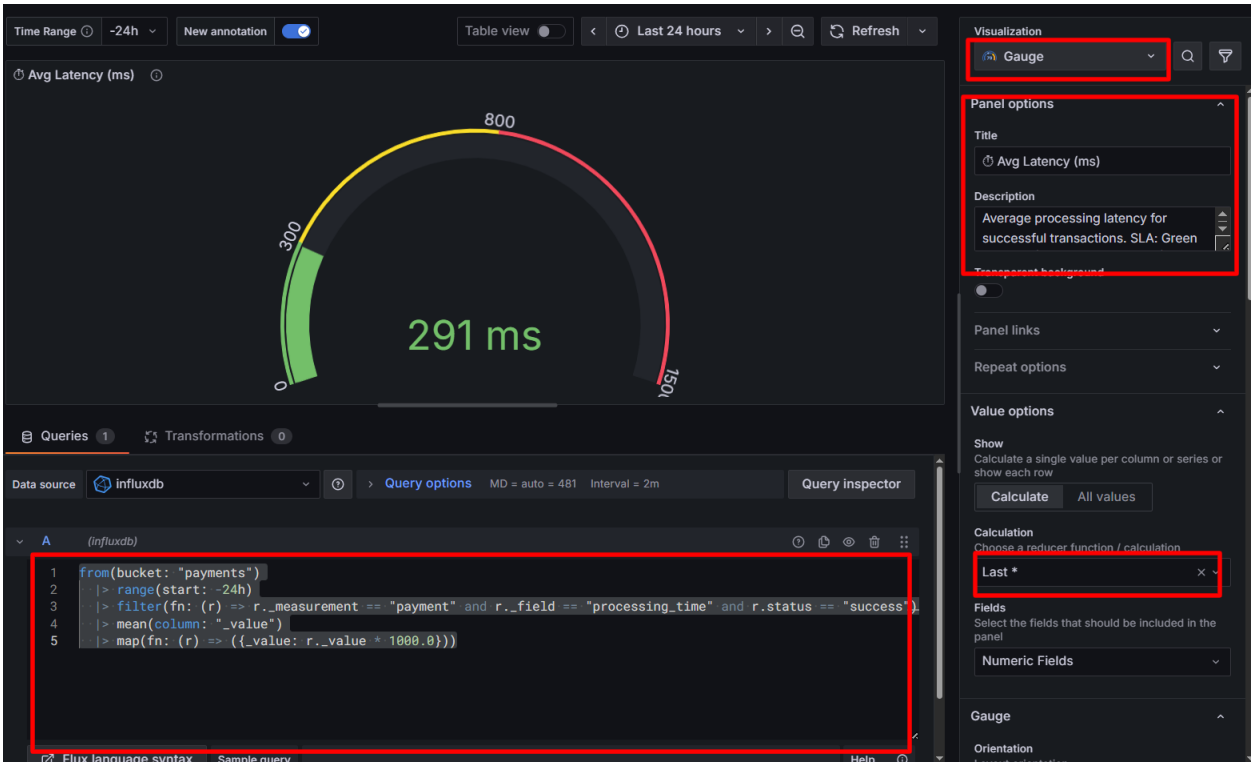
```
 from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) => r._measurement == "payment" and r._field == "amount")
  |> group(columns: ["status"])
  |> count()
  |> group(columns: [])
  |> map(fn: (r) => ({r with _dummy: "total"}))
  |> pivot(rowKey: ["_dummy"], columnKey: ["status"], valueColumn: "_value")
  |> map(fn: (r) => ({
    _value: (float(v: r.success) / float(v: (r.success + r.failed))) * 100.0,
    _time: now()
  }))
```

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

⏱ Panel 3 : Latence Moyenne (ms)

Type : Gauge avec arc



Configuration GUI

Panel Settings

- **Panel Title** : `⏱ Avg Latency (ms)`
- **Panel Type** : `Gauge`
- **Dimensions** : Position x:12, y:0 | Taille h:8, w:6

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

- **Description** : `Average processing latency for successful transactions. SLA: Green <300ms | Yellow 300-800ms | Red >800ms. Impacts user experience`

Field Configuration

Standard Options

- **Unit** : `Milliseconds (ms)`
- **Decimals** : `0`
- **Min** : `0`
- **Max** : `1500`

Threshold Options

- **Mode** : `Absolute`
- **Steps** :
  - Step 1 : `0` → `green`
  - Step 2 : `300` → `yellow`
  - Step 3 : `800` → `red`

Panel Options

Gauge Type : `Arc` **Show Threshold Labels** : `ON` **Show Threshold Markers** : `ON`

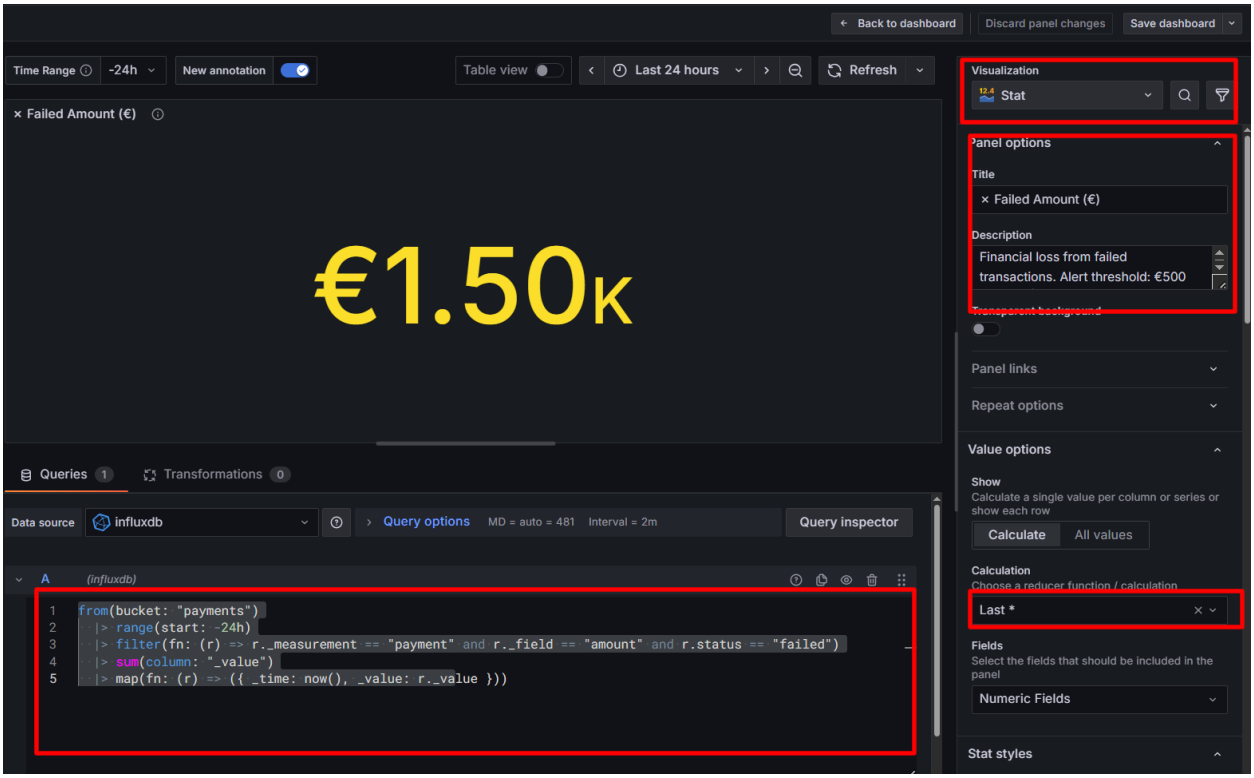
Requête InfluxDB (Flux)

```
from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) => r._measurement == "payment" and r._field == "processing_time" and r.status == "success")
  |> mean(column: "_value")
  |> map(fn: (r) => ({_value: r._value * 1000.0}))
```

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

❌ Panel 4 : Montant Échoué (€)



Type : Stat Card

Configuration GUI

Panel Settings

- **Panel Title** : ❌ `Failed Amount (€)`
- **Panel Type** : `Stat`
- **Dimensions** : Position x:18, y:0 | Taille h:8, w:6
- **Description** : `Financial loss from failed transactions. Alert threshold: €500 (Yellow) | €2000 (Red). Indicates system reliability issues`

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Field Configuration

Standard Options

- **Unit** : `currencyEUR`
- **Decimals** : `2`

Threshold Options

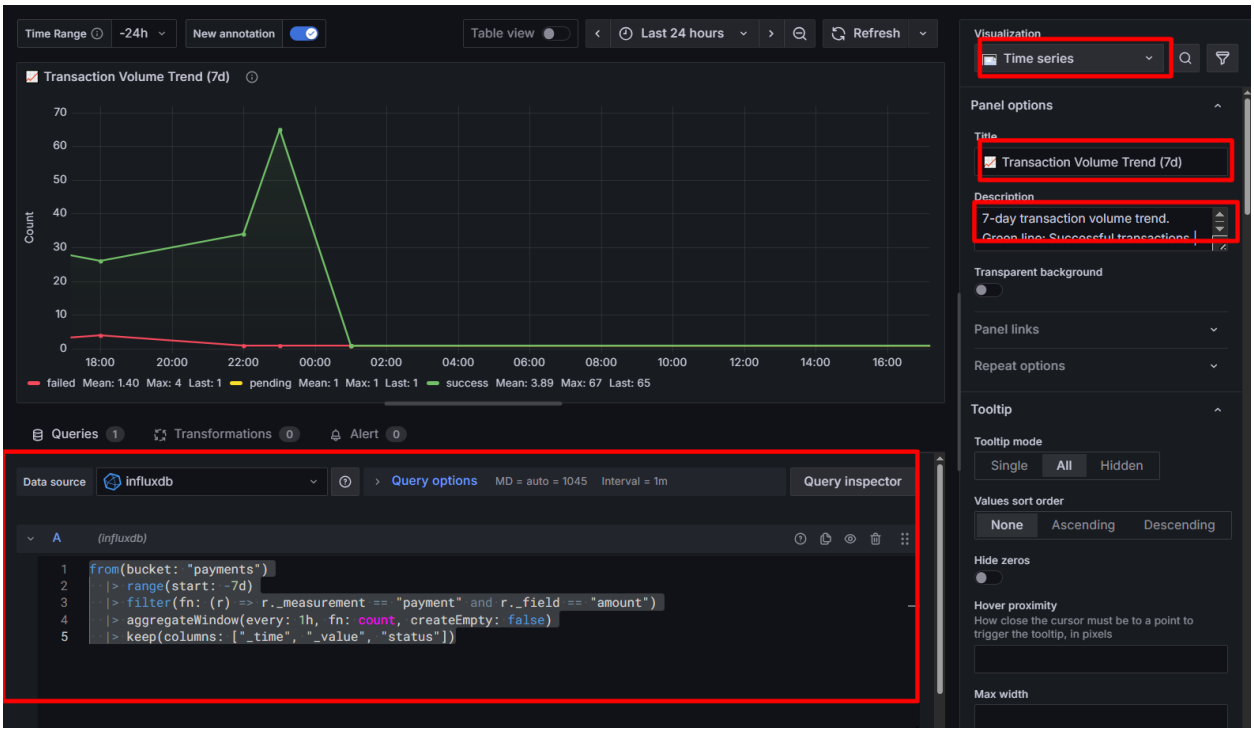
- **Mode** : `Absolute`
- **Steps** :
  - Step 1 : `0` → `green`
  - Step 2 : `500` → `yellow`
  - Step 3 : `2000` → `red`

Requête InfluxDB (Flux)

```
from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) => r._measurement == "payment" and r._field == "amount" and r.status == "failed")
  |> sum(column: "_value")
  |> map(fn: (r) => ({ _time: now(), _value: r._value }))
```

📈 Panel 5 : Tendance Volume (7j)

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com



Type : Time Series (Line Chart)

Configuration GUI

Panel Settings

- **Panel Title** : 📈 `Transaction Volume Trend (7d)`
- **Panel Type** : `Time series`
- **Dimensions** : Position x:0, y:8 | Taille h:10, w:12
- **Description** : `7-day transaction volume trend. Green line: Successful | Red line: Failed. Identify volume anomalies`

Field Configuration

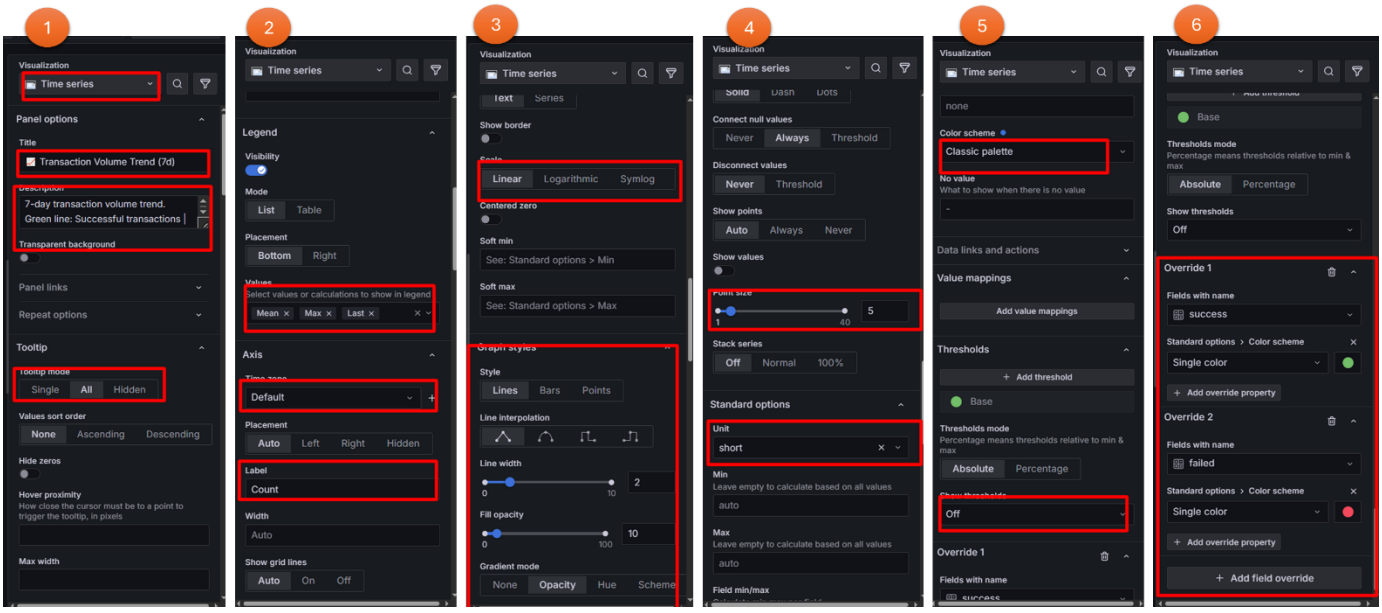
Standard Options

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

- **Unit** : `Short`
- **Custom → Line Interpolation** : `Linear`
- **Custom → Line Width** : `2`
- **Custom → Fill Opacity** : `10`
- **Custom → Show Points** : `Auto`

Overrides

- Override 1 : Nom `success` → Couleur `green`
- Override 2 : Nom `failed` → Couleur `red`



Panel Options

Legend :

- **Display Mode** : `List`
- **Placement** : `Bottom`
- **Calcs** : `Mean`, `Max`, `Last`
- **Show Legend** : `ON`

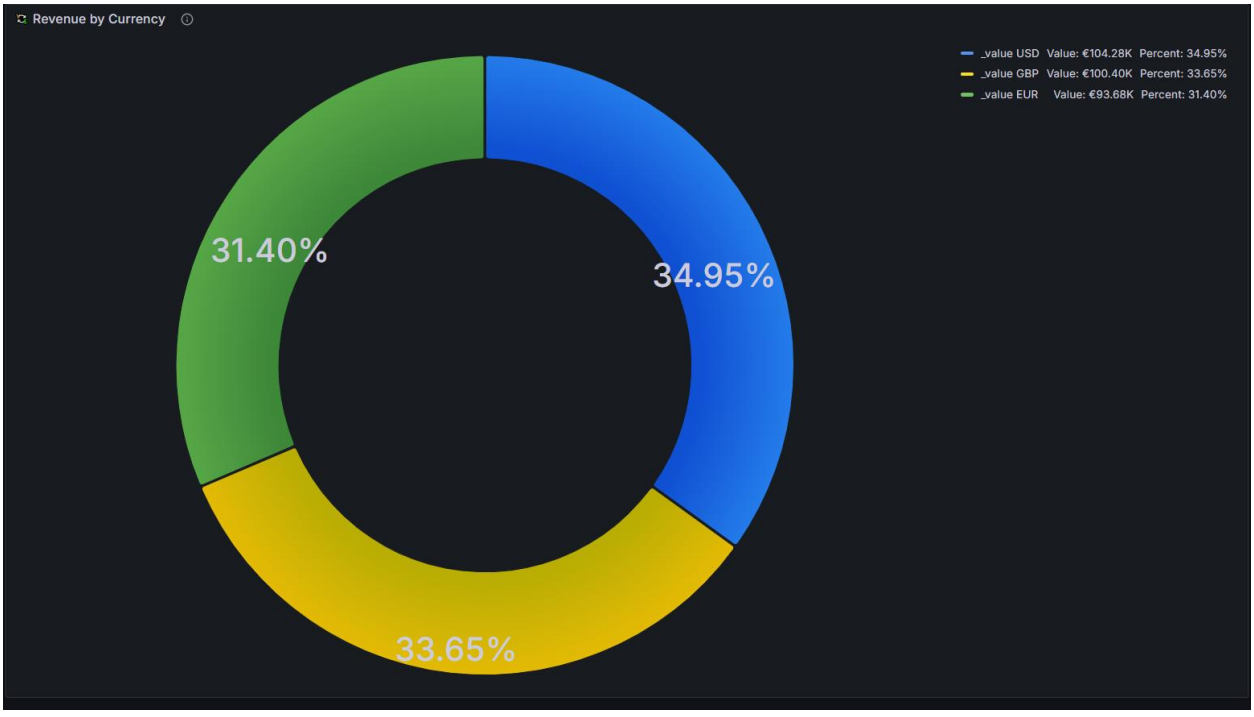
Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Tooltip : `Multi series`

Requête InfluxDB (Flux)

```
from(bucket: "payments")
  |> range(start: -7d)
  |> filter(fn: (r) => r._measurement == "payment" and r._field == "amount")
  |> aggregateWindow(every: 1h, fn: count, createEmpty: false)
  |> keep(columns: ["_time", "_value", "status"])
```

💱 Panel 6 : Revenue par Devise



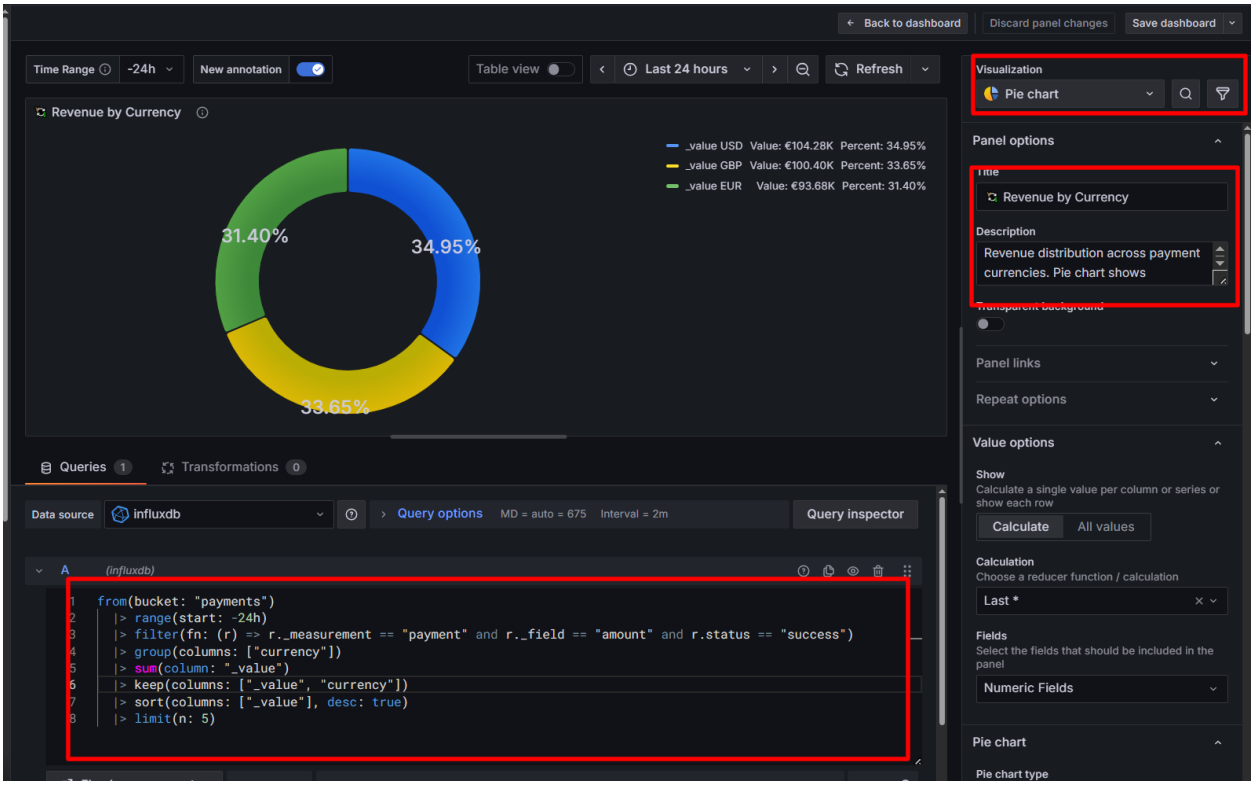
Type : Pie Chart

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Configuration GUI

Panel Settings



- **Panel Title** : `Revenue by Currency`
- **Panel Type** : `Pie chart`
- **Dimensions** : Position x:12, y:8 | Taille h:10, w:12
- **Description** : `Revenue distribution across payment currencies. EUR/USD/GBP/CHF/JPY split. Helps identify currency exposure`

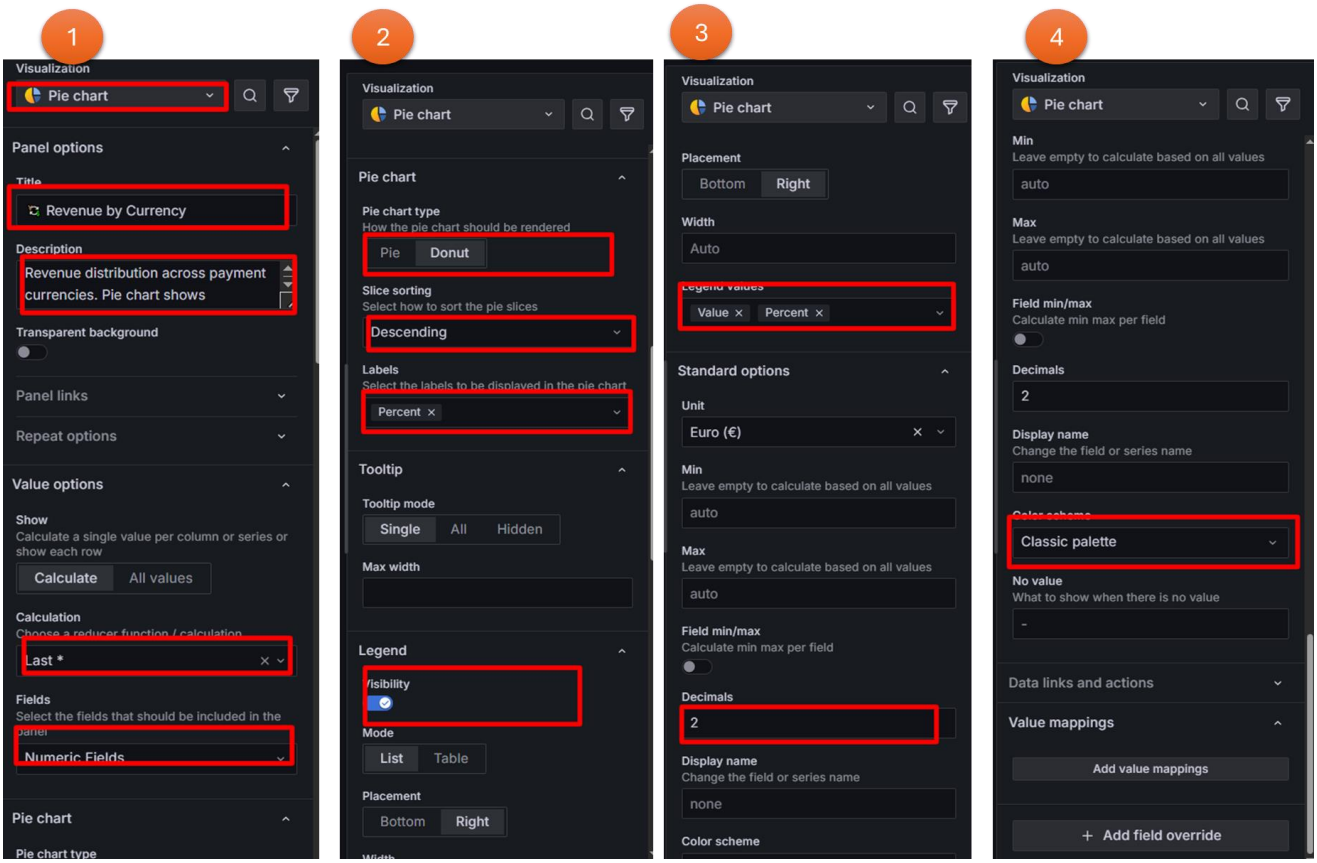
Field Configuration

Standard Options

- **Unit** : `currencyEUR`

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

- **Decimals** : `2`
- **Color Mode** : `Palette Classic`



Panel Options

Pie Type : `Donut` **Display Labels** : `Percent` **Legend** :

- **Display Mode** : `List`
- **Placement** : `Right`
- **Values** : `Value`, `Percent` **Sort** : `Descending`

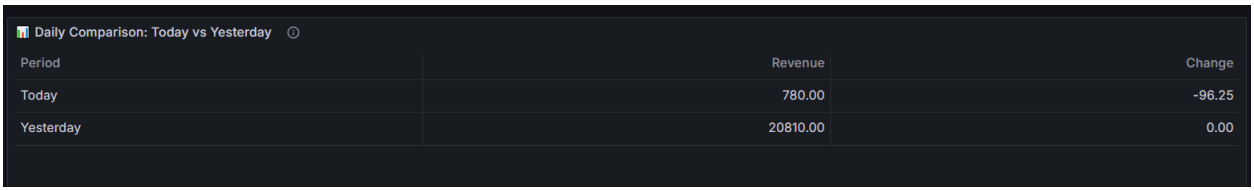
Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Requête InfluxDB (Flux)

```
from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) => r._measurement == "payment" and r._field == "amount" and r.status == "success")
  |> group(columns: ["currency"])
  |> sum(column: "_value")
  |> keep(columns: ["_value", "currency"])
  |> sort(columns: ["_value"], desc: true)
  |> limit(n: 5)
```

📊 Panel 7 : Comparaison Jour/Jour

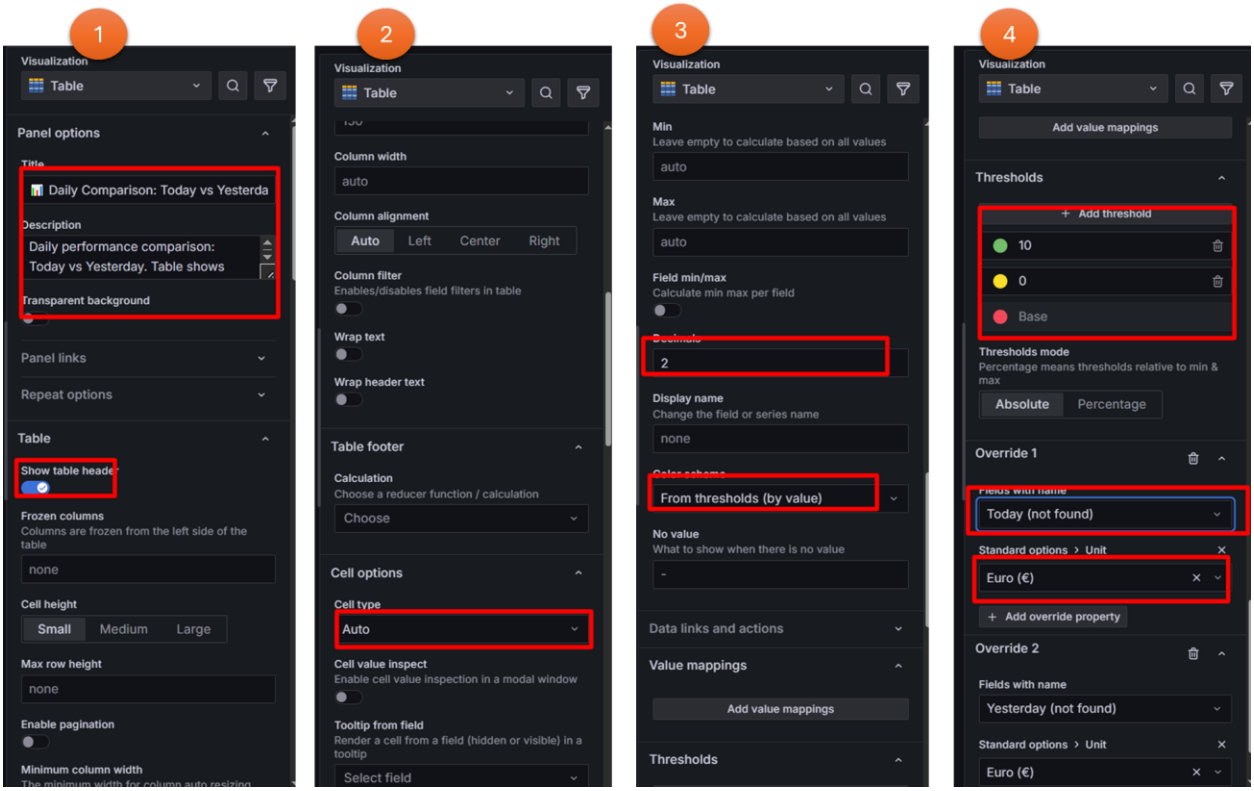


Type : Table

Configuration GUI

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com



Panel Settings

- **Panel Title** : 📊 `Daily Comparison: Today vs Yesterday`
- **Panel Type** : `Table`
- **Dimensions** : Position x:0, y:18 | Taille h:10, w:24
- **Description** : `Daily performance comparison with variation %`

Field Configuration

Standard Options

- **Decimals** : `2`

Overrides

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

- Override "Today" : Unit `currencyEUR`
- Override "Yesterday" : Unit `currencyEUR`

Panel Options

Cell Height : `sm` **Show Header** : `ON`

Requête InfluxDB (Flux)

```
 import "array"


todayData = from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) =>
    r._measurement == "payment" and
    r._field == "amount" and
    r.status == "success"
  )
  |> sum(column: "_value")
  |> findRecord(fn: (key) => true, idx: 0)

yesterdayData = from(bucket: "payments")
  |> range(start: -48h, stop: -24h)
  |> filter(fn: (r) =>
    r._measurement == "payment" and
    r._field == "amount" and
    r.status == "success"
  )
  |> sum(column: "_value")
  |> findRecord(fn: (key) => true, idx: 0)

// --- Defensive defaulting ---
todayValue = if exists todayData._value then float(v: todayData._value)
else 0.0
```

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

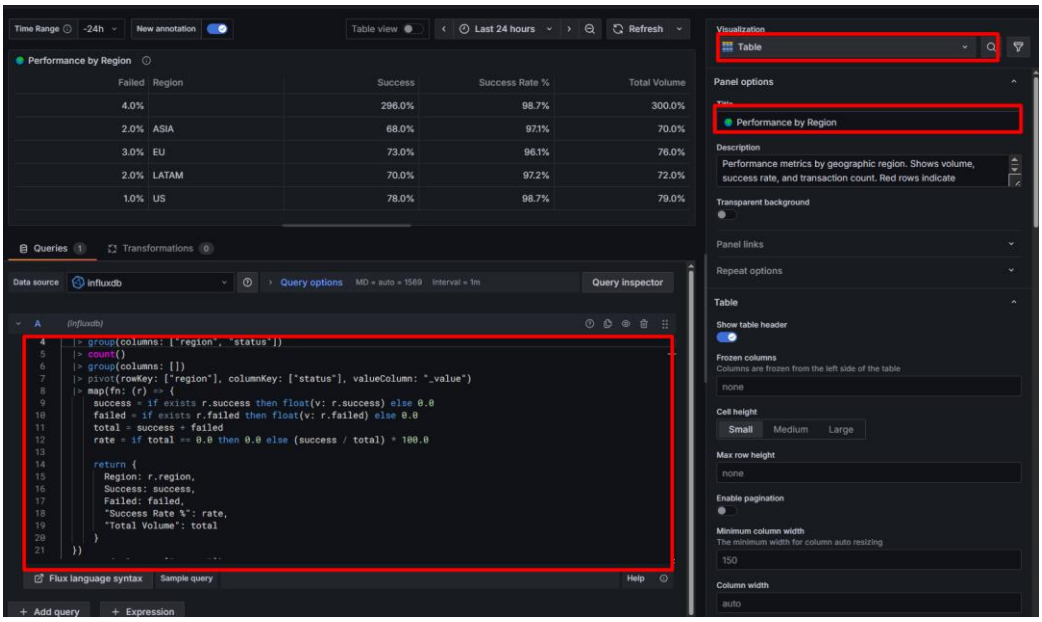
```
yesterdayValue = if exists yesterdayData._value then float(v:
yesterdayData._value) else 0.0

// --- Compute change safely ---
change = if yesterdayValue == 0.0 then 0.0
  else (todayValue - yesterdayValue) / yesterdayValue * 100.0

// --- Return a small summary table ---
array.from(rows: [
  {Period: "Today", Revenue: todayValue, Change: change},
  {Period: "Yesterday", Revenue: yesterdayValue, Change: 0.0}
])
```

🌎 Panel 8 : Performance par Région —

Grafana Panel Configuration

Panel Title: Performance by Region
Description: Performance metrics by geographic region. Displays volume, success rate, and transaction count. Red rows highlight underperforming regions requiring attention.

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com



🧮 KPI Formulas

- **Total Volume (V)** = `count(payment_id)`
- **Success Rate (%)** = `(success / total) * 100`
- **Failure Rate (%)** = `(failed / total) * 100`
- **Average Latency (ms)** = `mean(latency)`

🧩 InfluxDB Flux Query

```
from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) => r._field == "amount")
  |> group(columns: ["region", "status"])
  |> count()
```

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

```
  |> group(columns: [])
  |> pivot(rowKey: ["region"], columnKey: ["status"], valueColumn:
"_value")
  |> map(fn: (r) => {
      success = if exists r.success then float(v: r.success) else 0.0
      failed = if exists r.failed then float(v: r.failed) else 0.0
      total = success + failed
      rate = if total == 0.0 then 0.0 else (success / total) * 100.0

      return {
        Region: r.region,
        Success: success,
        Failed: failed,
        "Success Rate %": rate,
        "Total Volume": total
      }
  })
  |> sort(columns: ["Region"])
```

⚙️ Visualization Settings

- **Visualization Type:** Table
- **Transparent background:** ✅
- **Show header:** ✅
- **Pagination:** Enabled
- **Column width:** Auto (min width 150)
- **Decimals:** 1 for % fields, 0 for volume
- **Unit:** Percent (0–100) for success rate

🎨 Thresholds & Coloring

Metric	Base	Warning	Critical	Color Scheme

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Success Rate	95%	90%	<90%	Green → Yellow → Red
Latency (ms)	100	500	>1000	Green → Orange → Red

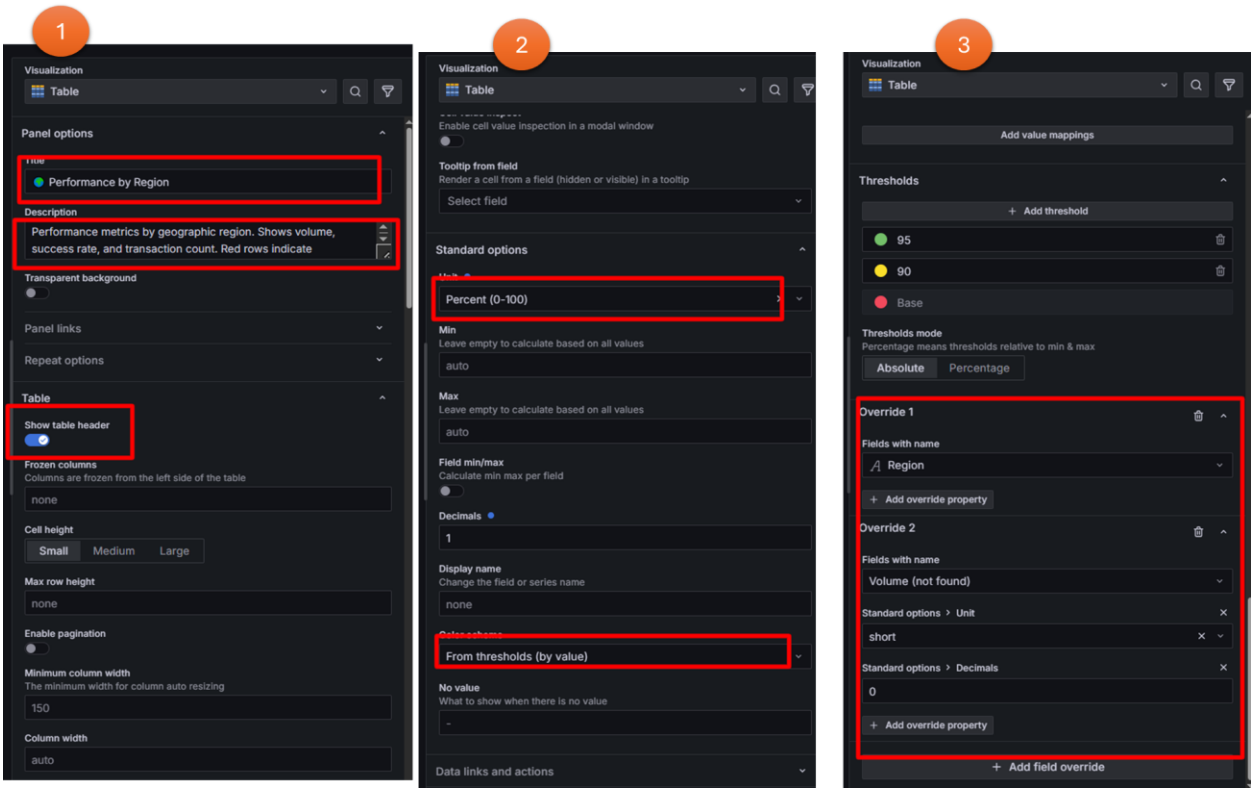
Rows where `success_rate < 90` are highlighted in **red** to indicate degraded regional performance.

Overrides

Field	Property	Value
Region	Text align	Left
Volume	Unit	short
Success Rate	Unit	percent
Success Rate	Decimals	1
Latency	Unit	ms

Tags: payments, performance , region, observability, SLA

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

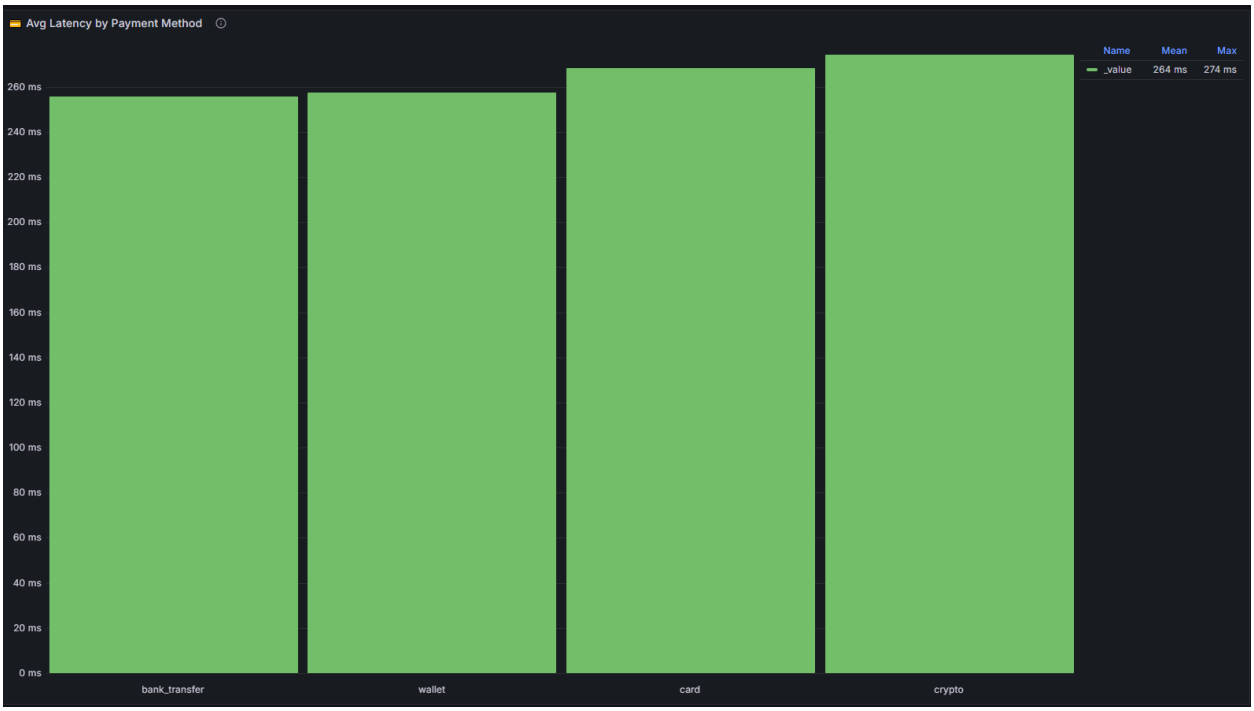


💳 Panel 9 : Latence par Méthode

Description :
Mesure les latences moyennes par méthode de paiement (carte, virement, wallet, etc.) sur les 24 dernières heures. Affiche les métriques de performance (P50, P95, P99) pour détecter les dégradations de SLA.

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com



📊 Requête Flux (InfluxDB)

```
from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) => r._field == "processing_time")
  |> group(columns: ["payment_method"])
  |> mean(column: "_value")
  |> group(columns: [])
  |> map(fn: (r) => ({
    _value: r._value * 1000.0,
    payment_method: r.payment_method
  }))
  |> sort(columns: ["_value"])
```

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

📊 Visualisation : Bar Chart

- **Type :** Bar chart
- **X-Axis :** `payment_method`
- **Y-Axis :** `_value` (latency in ms)
- **Orientation :** Verticale
- **Bar width :** 0.97
- **Bar radius :** 0
- **Show values :** Activé
- **Color by field :** Automatique (basé sur les seuils)
- **Tooltip :** Affiche les valeurs *Mean* et *Max*

⚙️ Options du panneau

- **Title :** 💳 Avg Latency by Payment Method
- **Description :** Latency percentiles: P50 (median), P95 (95th percentile), P99 (tail latency). Detects SLA violations and system performance degradation. Red P99 >1000ms indicates performance issue.
- **Transparent background :** Oui
- **Unit :** milliseconds (ms)
- **Decimals :** 0
- **Legend :** Afficher Mean et Max
- **Auto refresh intervals :** 30s, 1m, 5m, 10m

🚦 Seuils (Thresholds)

Couleur	Valeur (ms)	Interprétation
🟢 Vert	< 500 ms	Performance optimale
🟡 Jaune	500–1000 ms	Attention, latence modérée

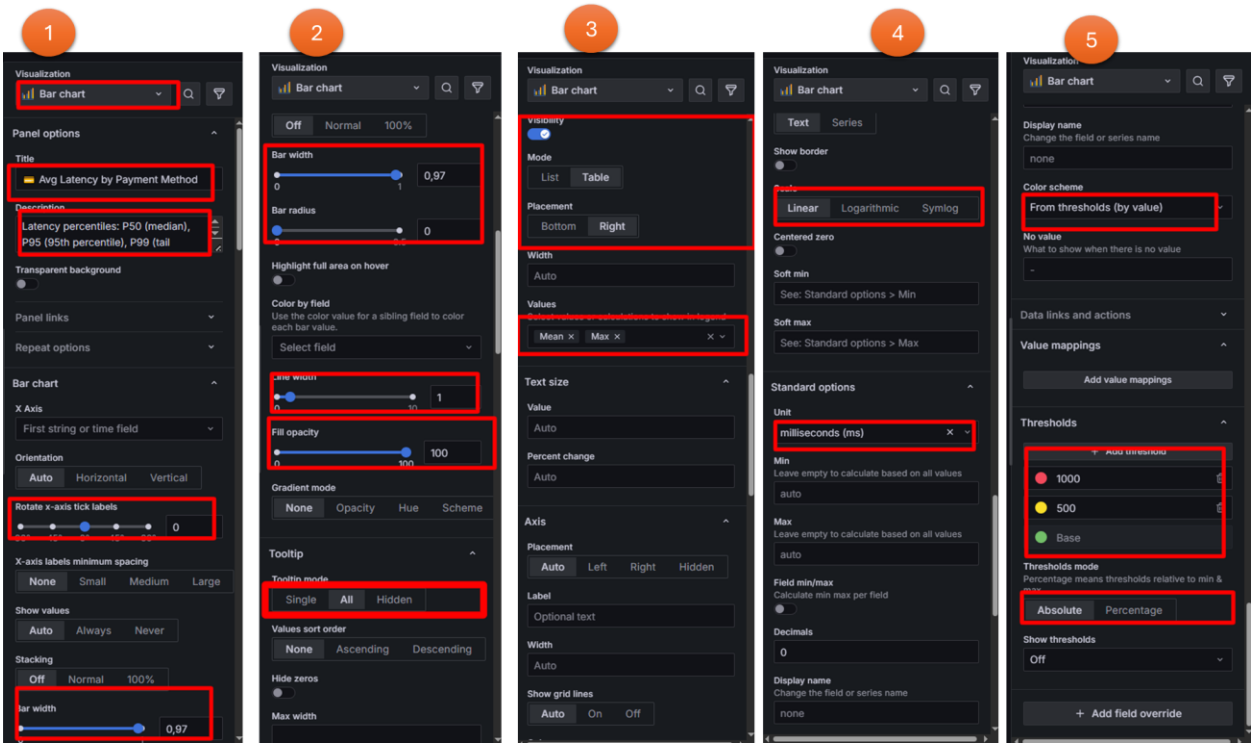
Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Couleur	Valeur (ms)	Interprétation
🔴 Rouge	> 1000 ms	Violation SLA / dégradation

🔁 Options supplémentaires

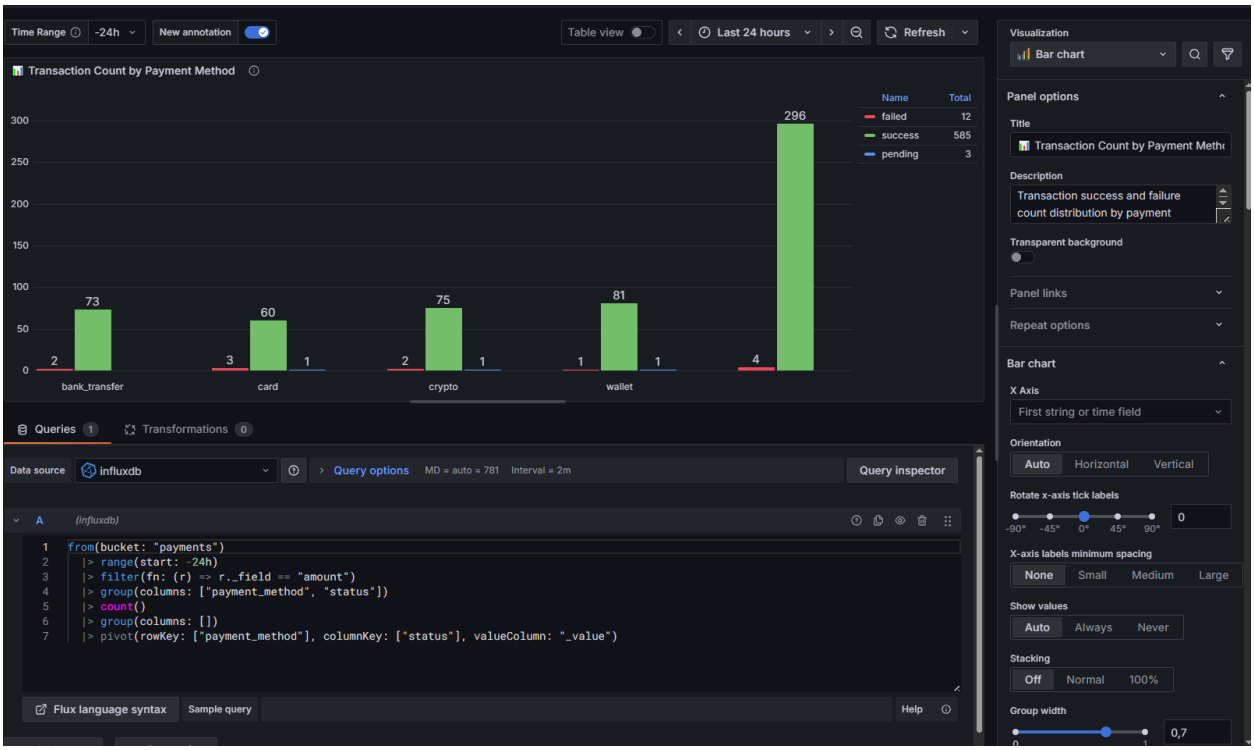
- **Repeat options :** Aucune (unique par méthode)
- **Transformations :** 0 (valeurs directes du Flux)
- **Datasource :** InfluxDB (bucket `payments`)
- **Interval d'échantillonnage :** 2m



📊 Panel 10 : Transactions par Méthode

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com



Type : Bar Chart

Configuration GUI

Panel Settings

- **Panel Title** : 📊 `Transaction Count by Payment Method`
- **Panel Type** : `Bar chart`
- **Dimensions** : Position x:12, y:38 | Taille h:10, w:12

Field Configuration

Color Mode : `Palette Classic` **Overrides** :

- `success` ⟶ `green`
- `failed` ⟶ `red`

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Panel Options

Bar Width : `0.97` **Stacking** : `None` **Legend** : `Table` | Right **Calcs** : `Sum`

Requête InfluxDB (Flux)

```
from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) => r._field == "amount")
  |> group(columns: ["payment_method", "status"])
  |> count()
  |> group(columns: [])
  |> pivot(rowKey: ["payment_method"], columnKey: ["status"], valueColumn:
"_value")
```

💪 Panel 11 : Performance Marques Cartes

Brand	Failed	Success	Success Rate %	Total Transactions
AMEX	1.0%	73.0%	98.6%	74.0%
DISCOVER	3.0%	75.0%	96.2%	78.0%
MASTERCARD	2.0%	76.0%	97.4%	78.0%
VISA	4.0%	91.0%	95.8%	95.0%

🏢 Card Brand Performance — Grafana Panel Configuration (Professional Edition)

Panel Type: Table
Data Source: InfluxDB (Bucket: payments)
Time Range: Last 24 hours (-24h)

🎯 Objective

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Visualize success rate per card brand (e.g., VISA, Mastercard, AMEX, Discover) to evaluate the reliability of different payment networks. This panel supports quick SLA verification and brand-level performance benchmarking.

🧮 KPI Formula

$$\text{Success Rate (\%)} = \frac{\text{Success Transactions}}{\text{Total Transactions}} \times 100$$

💾 Flux Query

```
from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) => r._field == "amount")
  |> group(columns: ["card_brand", "status"])
  |> count()
  |> group(columns: [])
  |> pivot(rowKey: ["card_brand"], columnKey: ["status"], valueColumn:
"_value")
  |> map(fn: (r) => {
      success = if exists r.success then float(v: r.success) else 0.0
      failed = if exists r.failed then float(v: r.failed) else 0.0
      total = success + failed
      rate = if total == 0.0 then 0.0 else (success / total) * 100.0

      return {
        Brand: r.card_brand,
        Success: success,
        Failed: failed,
        "Success Rate %": rate,
        "Total Transactions": total
```

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

```
    }
})
|> sort(columns: ["Brand"])
```

⚙️ Visualization Settings

- **Type: Table**
- **Show Header:** ✅
- **Transparent background:** ✅
- **Column Width: Auto**
- **Pagination: Enabled**
- **Minimum Column Width: 150 px**
- **Decimals: 1**
- **Unit: Percent (0–100)**
- **Color Scheme: From thresholds (by value)**

🎨 Threshold Configuration

Color	Threshold	Meaning
🟢 Green	> 95%	Excellent brand reliability
🟡 Yellow	90–95%	Acceptable but needs monitoring
🔴 Red	< 90%	Performance degradation — action required

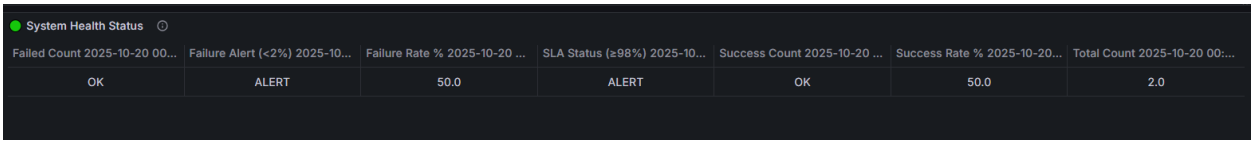
🧭 Overrides

- **Field name: Brand → Label for card type (VISA, Mastercard, etc.)**
- **Field name: success_rate → Numeric field used for threshold-based coloring.**

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

🟢 Panel 12 : Santé Système



Type : Table

Configuration GUI

Panel Settings

- **Panel Title** : 🟢 `System Health Status`
- **Panel Type** : `Table`
- **Dimensions** : Position x:0, y:58 | Taille h:8, w:24
- **Description** : `Real-time alerts: Success Rate SLA (target 98%), Latency SLA (target <300ms), Failure Rate (target <2%)`

Field Configuration

Value Mappings :

- `0` ⟶ `ALERT` (rouge)
- `1` ⟶ `OK` (vert)

Panel Options

Cell Height : `sm` **Show Header** : `ON`

Requête InfluxDB (Flux)

```
from(bucket: "payments")
  |> range(start: -24h)
  |> filter(fn: (r) => r._field == "amount")
  // 1 Compute counts by status per hour
```

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

```
  |> aggregateWindow(every: 1h, fn: count, createEmpty: false)
  |> group(columns: ["_time", "status"])
  // 2 Pivot so each row has success + failed for the same hour
  |> pivot(rowKey: ["_time"], columnKey: ["status"], valueColumn:
"_value")
  // 3 Compute metrics safely
  |> map(fn: (r) => {
      success = if exists r.success then float(v: r.success) else 0.0
      failed = if exists r.failed then float(v: r.failed) else 0.0
      total = success + failed
      successRate = if total == 0.0 then 0.0 else (success / total) *
100.0
      failureRate = if total == 0.0 then 0.0 else (failed / total) * 100.0
      slaStatus = if successRate >= 98.0 then 1 else 0
      failureAlert = if failureRate < 2.0 then 1 else 0

      return {
        _time: r._time,
        "Success Count": success,
        "Failed Count": failed,
        "Total Count": total,
        "Success Rate %": successRate,
        "Failure Rate %": failureRate,
        "SLA Status (≥98%)": slaStatus,
        "Failure Alert (<2%)": failureAlert
      }
  })
```

🧭 Section 6 : Configuration des Alertes (Email & Slack)

🎯 Objectif

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Mettre en place un système d'alerte intelligent dans Grafana pour notifier automatiquement les incidents de performance des paiements (taux de réussite < 95% ou latence > 2s) via **email** et **Slack Webhook**.

🔗 Étape 1 : Accéder à la configuration des alertes

1. Connectez-vous à **Grafana** (port 3000).
2. Dans la barre latérale gauche, cliquez sur **Alerting → Contact points**.
3. Cliquez sur **+ Add contact point**.

📩 Étape 2 : Créer un canal d'alerte Email

1. **Name** : `Email_Alert_Payments`
2. **Type** : Email
3. **Addresses** : `ops_team@bankinglab.local` (ou votre adresse de test)
4. Cliquez sur **Save contact point**.

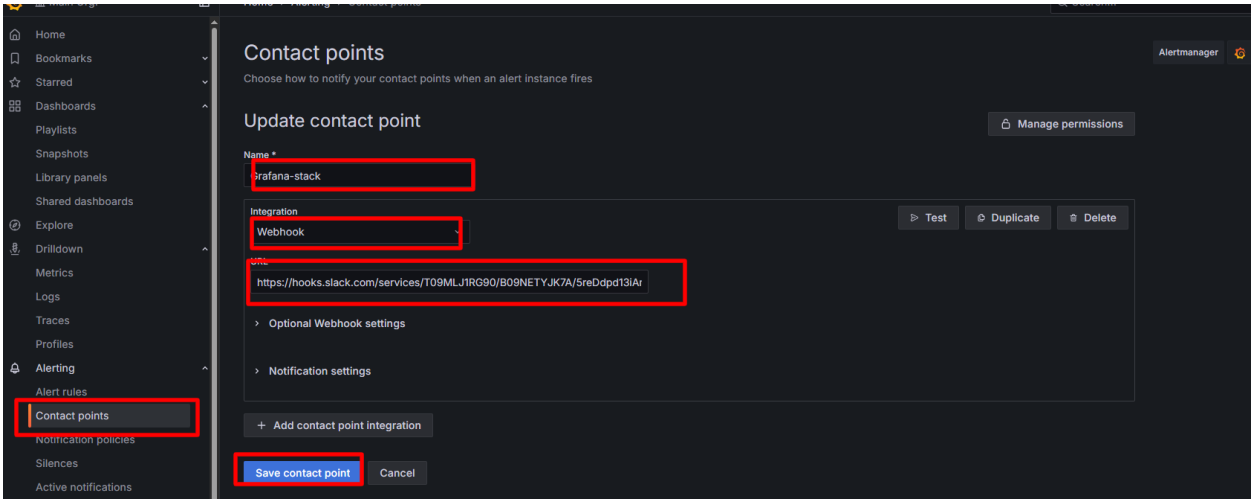
🔍 **Validation** : Vous pouvez cliquer sur **Test** pour envoyer un email de test.

💬 Étape 3 : Créer un canal d'alerte Slack

1. Cliquez sur **+ Add contact point**.
2. **Name** : `Slack_Alert_Payments`
3. **Type** : Slack
4. **Webhook URL** : Collez votre webhook Slack (exemple : `https://hooks.slack.com/services/T000/B000/XXXX`)
5. **Message Template** :
6. 🚨 `*Alerte Paiement détectée !*`
7. `• Panel : {{ $labels.panel }}`
8. `• Condition : {{ $labels.condition }}`
9. `• Valeur : {{ $values }}`

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

10. • Heure : {{ $labels.time }}
11. Cliquez sur **Save contact point**.



⚙️ Étape 4 : Créer une règle d'alerte (Alert Rule)

1. Allez dans **Alerting → Alert rules → + New alert rule**.
2. **Name** : `Payment_SLA_Alert`
3. **Folder** : `Observability_Payments`
4. **Data source** : InfluxDB
5. **Query (Flux)** :

```
from(bucket: "payments")
  |> range(start: -5m)
  |> filter(fn: (r) => r._field == "success_rate" or r._field ==
"avg_latency")
  |> aggregateWindow(every: 1m, fn: mean)
  |> yield(name: "mean")
```

6. **Condition** :
7. `WHEN success_rate < 95 OR avg_latency > 2`
8. **Evaluate every** : 1m
9. **For** : 2m (évite les fausses alertes)

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

10. **Contact point** : sélectionnez `Slack_Alert_Payments` et `Email_Alert_Payments`.
11. **Message customisé** :

```
⚠️ *Alerte Observabilité Paiement*
Taux de réussite : {{ $values.success_rate }}%
Latence moyenne : {{ $values.avg_latency }}s
```

📊 Étape 5 : Relier l'alerte au Dashboard

1. Ouvrez votre dashboard **Observabilité Paiements**.
2. Sélectionnez le panneau 🏢 `Card Brand Performance` ou ⏱️ `Latence Moyenne`.
3. Cliquez sur **Edit → Alert → Create Alert Rule**.
4. Liez à la règle `Payment_SLA_Alert`.
5. Sauvegardez le dashboard.

🔔 Étape 6 : Tester et valider les notifications

1. Simulez une baisse du taux de réussite (ex. < 90%) ou augmentez la latence.
2. Observez l'alerte apparaître dans **Alerting → Alert rules** avec l'état **Firing**.
3. Vérifiez la réception du message dans :
   - **Slack** (canal `#payment-alerts`)
   - **Email** (dossier réception)

✨ Conseils Finaux

1. **Cohérence des Couleurs** : Maintenir green=OK, yellow=warning, red=critical
2. **Refresh Rate** : Recommandé `30s` à `1m` pour production
3. **Alerting** : Configurer notification via email pour SLA < 98%
4. **Performance** : Limiter requêtes à -24h pour dashboard principal
5. **Exportation** : Dashboard JSON disponible pour CI/CD Terraform

---

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production

Formation: Grafana
Formateur: Mokhtar Sellami
Mail: mokhtar.sellami@data2-ai.com

Version : `v1.0` | **Dernière mise à jour** : mardi 28 octobre 2025| **Statut** : ✅ Production