



Deynao

Oracle supervision and diagnostics

Deynao — Autonomous supervision

Watch your databases continuously and be alerted automatically

Audience — Administrator · Operator (DBA)

Level — Intermediate — the Getting Started guide is a recommended prerequisite

Guide version — 1.1

Deynao — Autonomous supervision

In the *Getting Started* guide, you supervised a database **on demand**: you open the dashboard, Deynao connects and shows the state. **Autonomous supervision** goes further: Deynao watches your databases **continuously**, in the background, and **alerts you automatically** — even when no one has the tool open.

This guide shows how to enable and operate it, **safely**: a dedicated account, a non-intrusive collector, **transparent alert rules**, **audited** email alerts, a **daily report**, maintenance windows, and fine-grained access control.

 **Prerequisites.** Having read the *Getting Started* guide, and having a **dedicated supervision account** (we get to it). The activation and configuration described here require the **Administrator** or **Operator** role.

Step 1 — The three usage modes

Before enabling anything, you must distinguish **three ways** of using Deynao — never to be confused:

Mode	Footprint	When to use it
One-off diagnostic	None (nothing is kept)	A quick look, leaving no trace
Database in the fleet, on demand	The connection is kept (encrypted); no background collection	You open the dashboard whenever you want
Autonomous supervision	Connection kept + collector active	Deynao watches continuously and alerts — the subject of this guide

 Autonomous supervision is **opt-in**: by default, a database added to the fleet is in "**on-demand**" mode. You enable it explicitly, database by database. Once active, the database's card offers "**View collected state**" (the last measured state, **without** a new connection) in addition to the "**Live dashboard**".

Step 2 — The dedicated supervision account

To watch **continuously**, Deynao needs an Oracle account. The **best practice** — and Deynao's firm recommendation — is to use a **dedicated, READ-ONLY account** (least privilege), rather than `sys` or an

application account.

Deynao **provides the script** to create this account, ready to run, right from the welcome screen ("View the *GRANT script*") as well as in the connection form.

deynao_grant_privileges_template.sql  Copy  Download

```
-- =====
-- Deynao - Dedicated supervision account (creation + read-only privileges)
-- =====
-- This script CREATES the account (if missing) then grants it CREATE SESSION
-- + read-only SELECT on system views. No write privilege, no DBA role, no access
-- to application data. Generic (no account/server/IP hardcoded).
-- Compatible with Oracle 11g / 12c / 19c / 21c.
--
-- HOW TO RUN: connect as SYS, then run this file. The script prompts for the
-- supervision account name + a password (masked), creates the account if needed
-- and grants the privileges. Safe to re-run.
-- - Windows : Command Prompt on the Oracle server, then
```

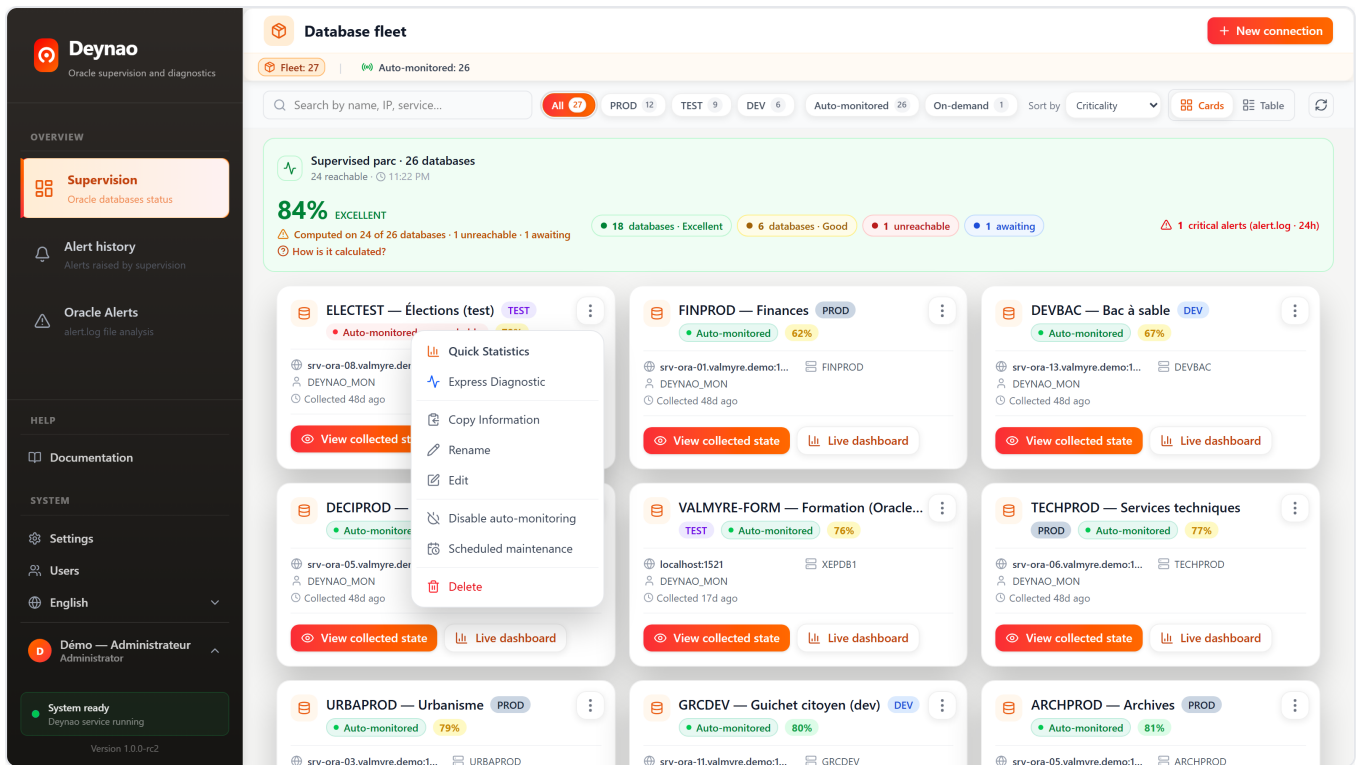
- The script **creates the account** (if it does not exist) and grants it the strict minimum: `CREATE SESSION` + `SELECT` **read-only** on system views.
- **No** write privilege, **no** DBA role, **no** access to application data.
- **Generic** (no name/IP hardcoded) and **compatible with Oracle 11g / 12c / 19c / 21c**.

 **Why a dedicated account?** It is **safer** (no risk of writes, minimal scope), **traceable** (Deynao's connections are identifiable), and **sufficient** for all supervision. An incident on this account never exposes your business data.

 **Step-by-step procedure** — get the script, run it, **verify** the privileges obtained and connect the account in Deynao: **The supervision account** guide.

Step 3 — Enable automatic supervision

Once the database is in the fleet, open its **actions menu** (:) and enable **automatic monitoring**.



On activation, Deynao **first tests the connection** — and this is where an essential safeguard comes in:

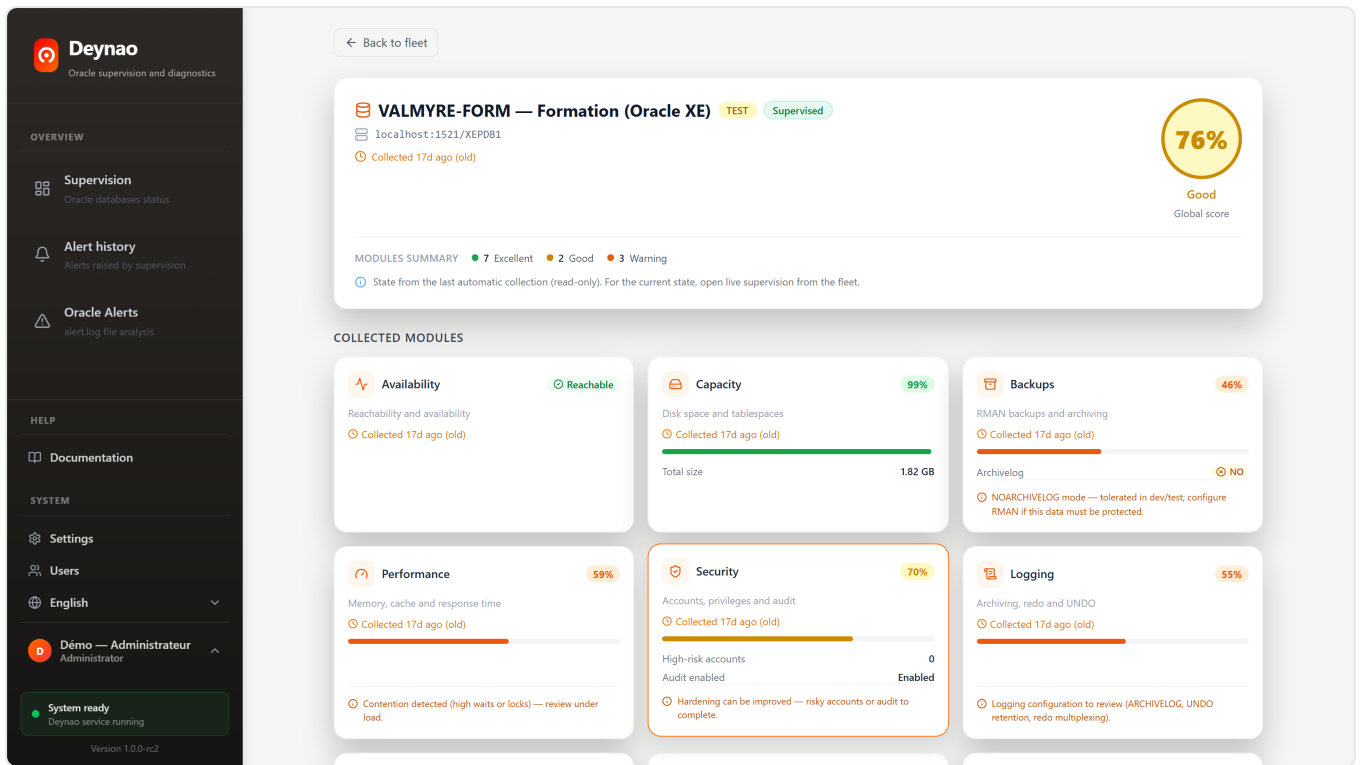
⚠ **Protected activation.** If the password is wrong, Deynao **refuses** activation and **arms a circuit-breaker**: it does **not** retry in a loop. Result: **never** an Oracle account locked by repeated attempts. If the database is simply **unreachable** (network), activation is accepted and Deynao will retry later, without hammering the listener.

Once enabled, the database switches to **"Auto-monitored"** (the fleet counter reflects it), and the **collector** starts watching it in the background.

Step 4 — The collector: background supervision

The **collector** is the heart of autonomous supervision. It runs **in the background**, domain by domain (the **same 14 domains** as the live dashboard), at regular intervals — **without anyone having to open the tool**.

The result of each pass is viewable **without a new connection**, via **"View collected state"**: this is the state **already measured** by the collector, timestamped.



Its design is entirely guided by the principle "*never harm the database*":

- **Protected connections** — an **authentication circuit-breaker** (never a locked account) and a **network back-off** (never a hammered listener) frame every connection.
- **Cancel watchdog** — on Oracle 11g in particular, a query that drags on is **cancelled** cleanly: the collector never **hangs**.
- **Read-only and zero pollution** — it **modifies nothing** and **writes nothing** into the `alert.log` of supervised databases.

💡 **Guaranteed consistency.** The collector computes scores with **exactly** the same logic as the live dashboard. The **Health Score** you see live is the one the collector uses to decide whether to **alert** you — no divergence.

🔧 **A clean pause on licensing.** If Deynao's **licence** is no longer operational (expired, missing), the collector **pauses** — no collection, alerts **frozen** — and **resumes on its own** as soon as it becomes valid again. No **artificial incident** piles up in the meantime.

Step 5 — Email alerts

This is the whole point of autonomous supervision: **being notified** without watching the screen.

5.1 Configure e-mail (SMTP)

Configuration is done once, in **Settings** → **E-mail**.

The screenshot shows the Deynao application interface. On the left is a dark sidebar with navigation links: Overview, Supervision, Alert history, Oracle Alerts, Help, Documentation, and System. The main content area is titled 'Settings' and 'E-mail'. It contains a form for configuring the SMTP server. The form includes a checkbox for 'Enable e-mail alerting', fields for 'Server' (localhost) and 'Port' (1025), fields for 'Username' (dba@client.example) and 'Password', a dropdown for 'TLS encryption' (Disabled), fields for 'Sender' (alertes@test.local) and 'E-mail language' (Français), and a field for 'Recipients' (moi@test.local). At the bottom are buttons for 'Test connection', 'Send a test e-mail', 'Cancel', and 'Save'.

- **Enable e-mail alerting** — the master switch.
- **Server · Port · TLS encryption · Username · Password** — your (internal, on-premise) SMTP server's coordinates.
- **Sender** and **recipients** — who sends, who receives.
- **E-mail language** — **independent** of the interface language: an English-speaking team can receive alerts in English even if the admin uses Deynao in French.

On-premise all the way. The SMTP server is **yours**. Alerts travel through **your** infrastructure — no external gateway, no leak.

5.2 Verify — without waiting for a real alert

Two buttons avoid bad surprises: **"Test connection"** (tests the SMTP dialogue) and **"Send a test e-mail"** (a real end-to-end e-mail). You then receive a clear, branded message — *"If you receive this message, Deynao's e-mail alerting configuration is working. No action is required: this is a simple send test."*

5.3 The alert e-mails

At the heart of alerting are the **alert e-mails** themselves — **educational**, written for an IT manager or a decision-maker as much as for a DBA: clear title, **problem**, **severity**, **since when**, then *"What it means"* and *"What to check"*, and finally the technical detail. Four moments punctuate an alert's life:

E-mail	When
New alert	The problem is confirmed (raised)
Reminder	It persists — re-notified at most once / 24 h
Escalation	It worsens (e.g. Warning → Critical)
Resolved	It has disappeared — with the real duration of the incident

💡 **The right tone for the right reader.** The e-mail explains the *why* in plain language (never needless jargon); and the **"Resolved"** e-mail does **not** re-explain the problem (it no longer exists) — it confirms, with the duration. Every word is useful.

5.4 The daily report

Beyond alerts, Deynao can send a **daily report**: a **summary** at a fixed time (Deynao state, fleet, alerts over the last 24 h). A **"Send report now"** button also allows an immediate **manual send**.

The screenshot shows the Deynao Settings interface. On the left is a dark sidebar with the Deynao logo and navigation links: OVERVIEW, Supervision, Alert history, Oracle Alerts, HELP, Documentation, SYSTEM, Settings (highlighted), Users, English, and Demo — Administrateur. The main content area is titled 'dba@client.example' and contains the following settings:

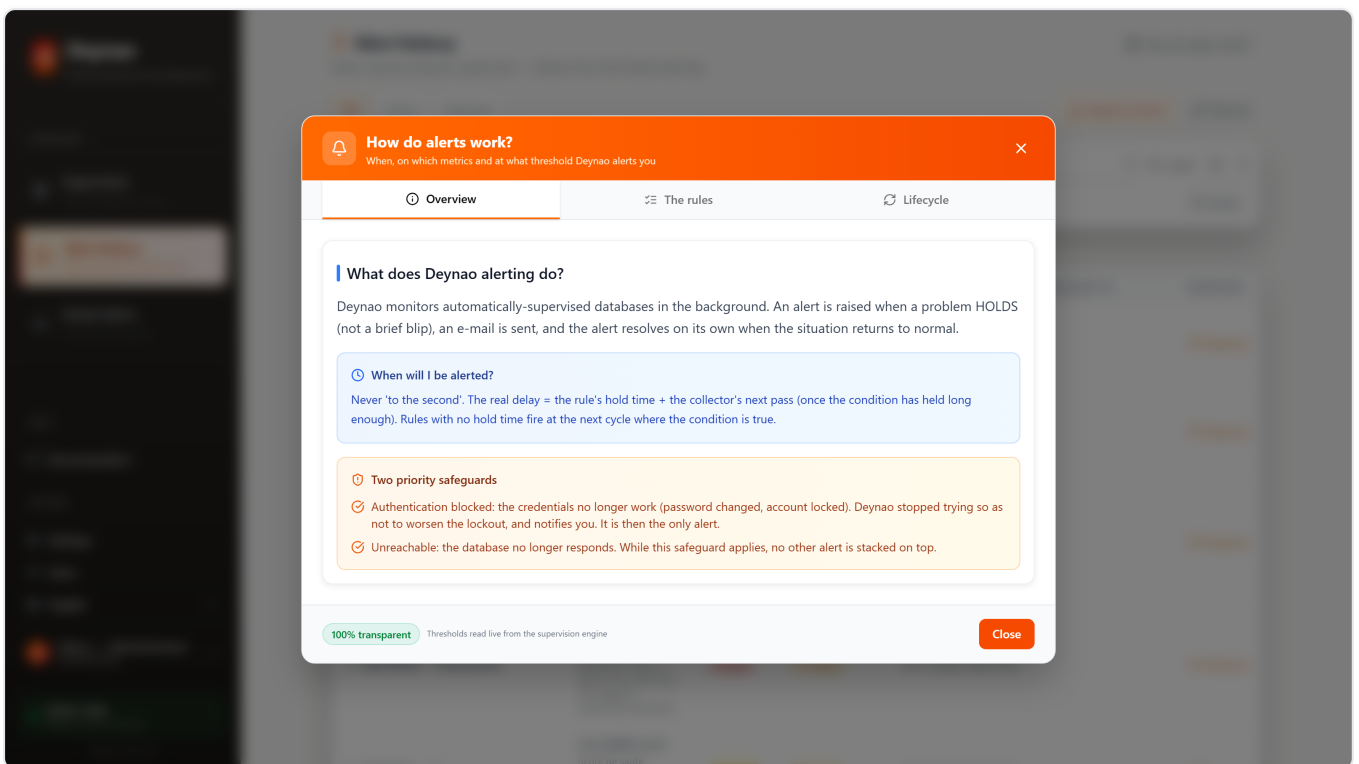
- TLS encryption:** A dropdown menu set to 'Disabled'. Below it, a note says: 'Automatic suits most servers; choose Required when a password is used.'
- Sender:** A text field containing 'alertes@test.local'.
- E-mail language:** A dropdown menu set to 'Français'. Below it, a note says: 'The e-mail language is independent from the interface language.'
- Recipients:** A text field containing 'moi@test.local'. Below it, a note says: 'Separate addresses with commas.'
- Daily e-mail report:** A section with a checkbox (unchecked) and a title 'Daily e-mail report'. Below the title is a description: 'One summary per day (Deynao status, fleet, alerts over the last 24 h) sent to the recipients above. If the report doesn't arrive as expected, go check Deynao.' Below this is a button 'Send the report now' and a note: 'Immediate send to the test recipient below — does not affect the scheduled send.'
- Send the test e-mail to:** A text field containing 'moi@test.local'. Below it, a note says: 'Test address remembered on this device — it is not added to the recipients.'
- A note below the test email field: 'The test buttons send an immediate request even when alerting is disabled — they only validate the configuration.'
- At the bottom of the settings area are three buttons: 'Test connection', 'Send a test e-mail', and 'Cancel'. To the right of these is a 'Save' button.
- At the very bottom is a 'License' section with a link icon and the text 'Deynao product license.'

The report you receive is **readable and actionable** — a one-sentence summary (*"In one sentence: 7 points require your attention: Critical tablespace on ...; Backup at risk on ...; Unreachable on ..."*), the **supervised fleet** (N databases · Excellent / Good / ... counters), each database's health, then the **current problems** in detail (named), the **last 24 h** (raised / resolved), and the **Deynao state** itself (health, version, uptime).

 **The daily report is also a "sign of life".** It includes Deynao's **own state** and the time of the **last collection**. Its closing sentence says it plainly: *"This report confirms that Deynao was operational when it was sent. **A missing report should alert you.**"** In other words: if the report **does not arrive**, Deynao (or the server) is down — and you know it.

Step 6 — How Deynao decides to alert (the rules)

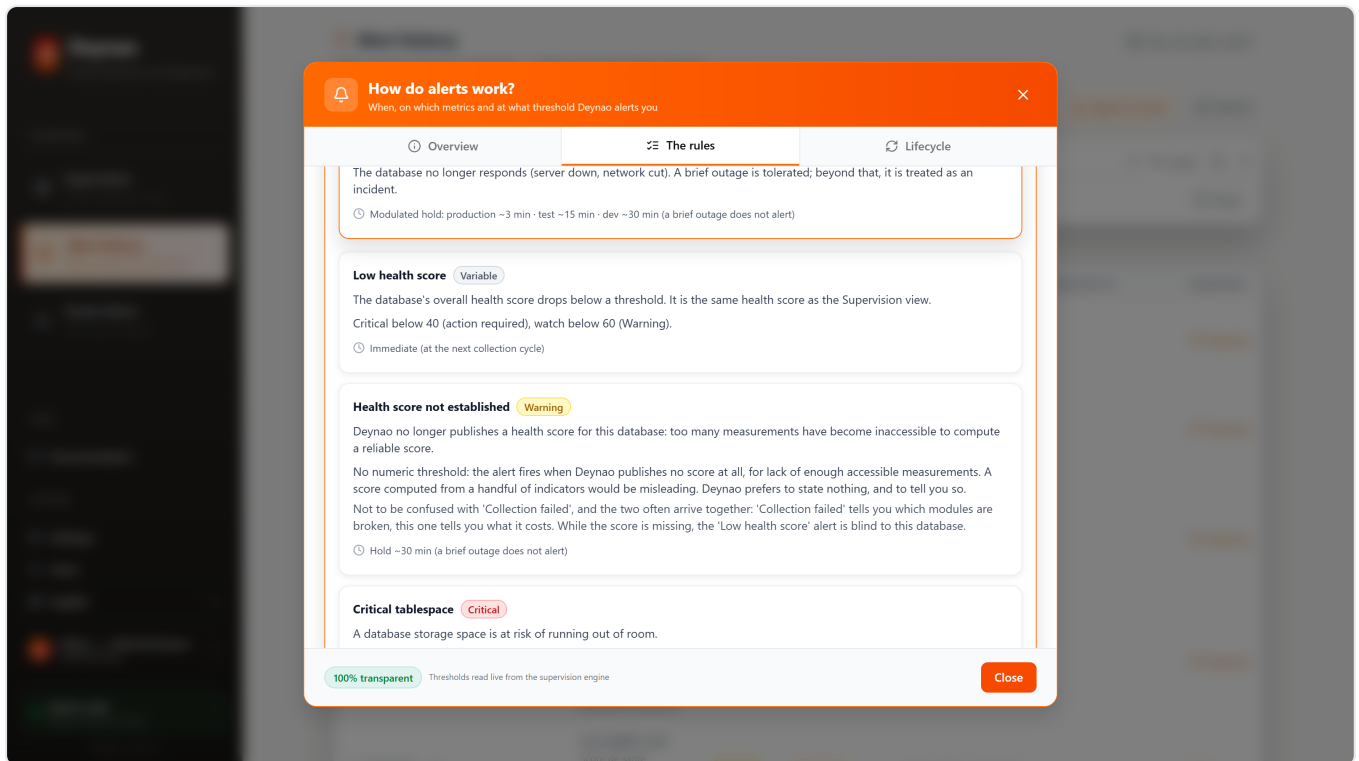
Good supervision alerts **neither too much nor too little**. Deynao makes its **rules fully transparent**: a **"How do alerts work?"** button (in the Alert history) opens the detail — and **all thresholds are read live from the engine**, no value hardcoded in a document.



Two common-sense principles:

- **An alert is raised when a problem *holds*** — not on every micro-blip. The real delay = the **rule's hold time** + the **collector's next pass**.
- **Two priority safeguards**. If **authentication is blocked** (password changed, account locked), Deynao **stops trying** (so as not to make it worse) and **that is the only alert**. If the database is **unreachable**, it does not **stack** any other alert on top.

6.1 The rules catalogue



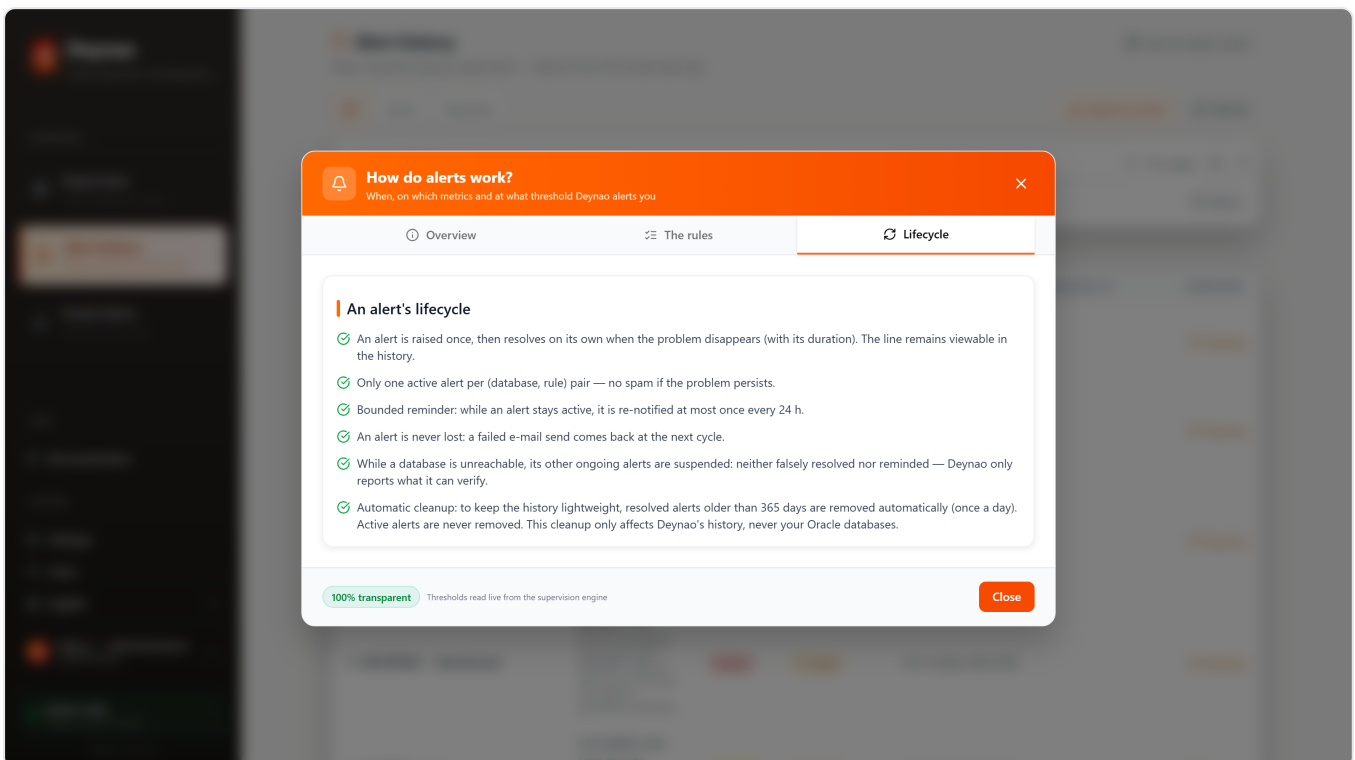
Rule	Severity	In short
Authentication blocked	Critical	Credentials that no longer work → Deynao stops trying and notifies
Unreachable	Critical	Database that no longer responds. Modulated hold time: prod ~3 min · test ~15 min · dev ~30 min
Low health score	Variable	Critical below 40 , watch below 60 (the same score as supervision)
Health score not established	Warning	Deynao no longer publishes a health score while the database responds: as long as it is missing, "Low health score" can no longer warn you
Critical tablespace	Critical	No % threshold: alerts only if the tablespace can no longer extend (autoextend maxed AND disk full)
Backup at risk	Variable	Last backup too old (threshold adapted to the rhythm: prod 2 d · test 8 d · dev 14 d) or ARCHIVELOG missing in prod
Recovery area full	Critical	FRA >90 % of non-reclaimable space (ORA-00257 risk) — production
Low security	Warning	Security score <30 or audit disabled — production
Scheduled job failure	Variable	Your Oracle jobs (payroll, accounting...) broken/failed — production
Collection failed	Warning	Part of the collection is durably failing while the database responds: partial data, not a dead database
Critical Oracle alerts	Warning	Critical codes found in the <code>alert.log</code>

🎯 **Thresholds that are *smart*, not mechanical.** A tablespace at 95 % but on autoextend **does not alert** (it can grow); a "full" FRA that is **reclaimable** neither. On the backup side, the same finesse: the age threshold **adapts to the real rhythm** (daily / weekly / monthly); a database in ARCHIVELOG whose **archives** are not backed up is flagged **even if the last "full" is recent** (RPO at risk); a production database that has **never** been backed up stands out immediately. Everywhere, Deynao alerts on the **real risk**, not on a raw percentage — fewer false positives, more trust.

🔔 **Two rules for the blind spot, and a criterion that matured.** "Collection failed" and "Health score not established" often arrive **together**, by design: the first **names the failing modules**, the second says **what it costs** (with no published score, the "Low health score" alert is blind). The "Collection failed" criterion is **temporal**: a module is declared durably down only if it is failing **and** has not collected successfully for at least **twice its cadence** (floor: **15 minutes**). Why not "three failures in a row"? Because with the collector's tight retries, a mere **transient hiccup** of a few minutes used to raise an alert that resolved right away: noise, the enemy of trust. The only exception: a module that has **never** collected successfully (missing privilege, absent view) is flagged from its very first failures.

Step 7 — An alert's lifecycle, and the History

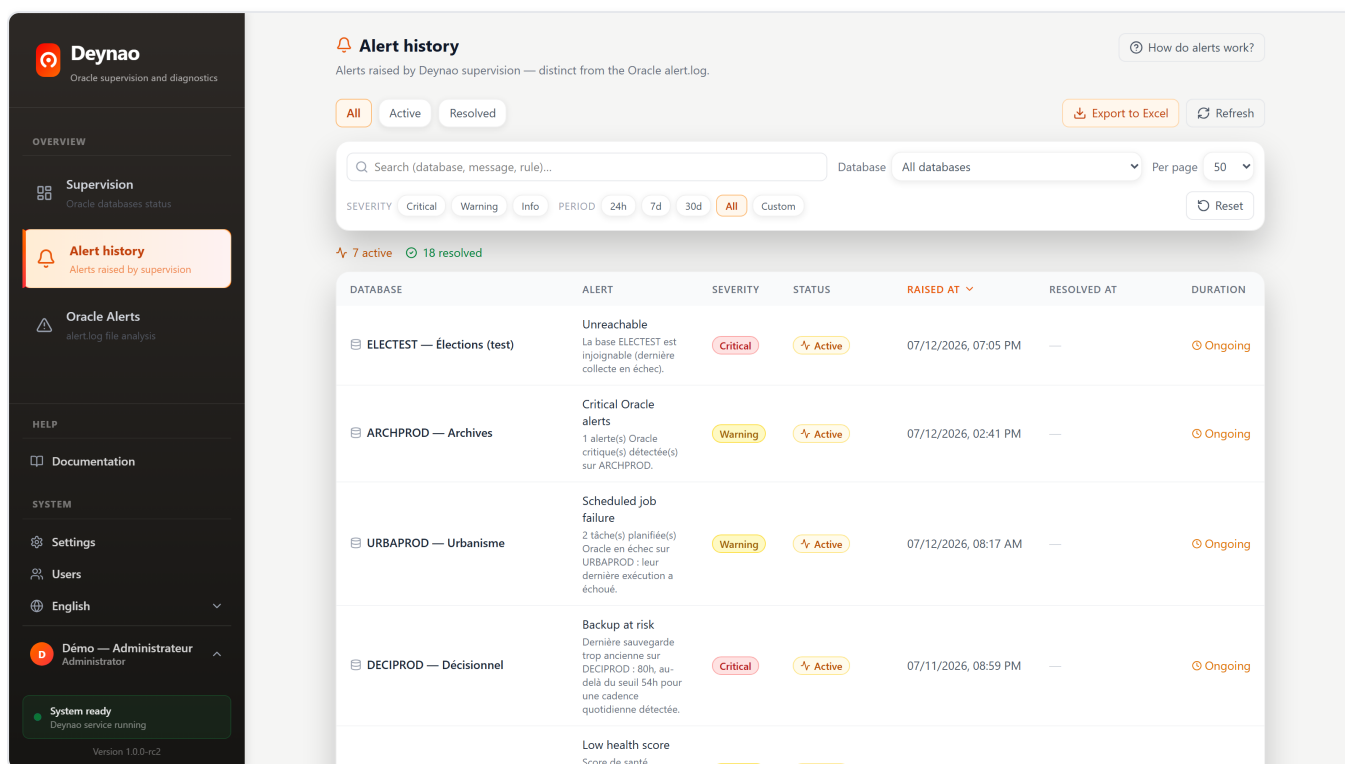
When the collector detects a problem that holds, it **raises an alert**, **sends the e-mail**, then **closes the alert** when the situation returns to normal. The **Lifecycle** tab of the same window details its guarantees:



- **A single active alert** per (*database, rule*) pair — **no spam** if the problem persists.
- **Bounded reminder** — an active alert is re-notified **at most once every 24 h**.
- **Never lost** — a failed e-mail **resurfaces** on the next cycle.
- **Cautious suspension** — while a database is unreachable, its other alerts are **suspended** (neither wrongly resolved nor reminded): Deynao only reports what it can **verify**.

 **Two subtleties that make the difference. Anti-oscillation** — Deynao waits for **continuous confirmation** before declaring "Resolved": an indicator that **oscillates** around a threshold (e.g. 59 % ↔ 61 %) **never** sends a misleading "Resolved" / "New alert" pair. **Escalation** — if a problem **worsens** (Warning → Critical), an **escalation** alert flags it, on the **same** incident line and **only once**; an improvement, on the other hand, generates **no** noise.

Everything is **recorded**. The **Alert history** lists every alert: database, severity, **status**, date **raised**, **resolved**, and **duration**.



Alert history
Alerts raised by Deynao supervision — distinct from the Oracle alert.log.

Buttons: All, Active, Resolved, Export to Excel, Refresh

Search: Search (database, message, rule)... Database: All databases Per page: 50

SEVERITY: Critical, Warning, Info PERIOD: 24h, 7d, 30d, All, Custom Reset

7 active 18 resolved

DATABASE	ALERT	SEVERITY	STATUS	RAISED AT	RESOLVED AT	DURATION
ELECTEST — Élections (test)	Unreachable La base ELECTEST est injoignable (dernière collecte en échec).	Critical	Active	07/12/2026, 07:05 PM	—	Ongoing
ARCHPROD — Archives	Critical Oracle alerts 1 alerte(s) Oracle critique(s) détectée(s) sur ARCHPROD.	Warning	Active	07/12/2026, 02:41 PM	—	Ongoing
URBAPROD — Urbanisme	Scheduled job failure 2 tâche(s) planifiée(s) Oracle en échec sur URBAPROD : leur dernière exécution a échoué.	Warning	Active	07/12/2026, 08:17 AM	—	Ongoing
DECIPROD — Décisionnel	Backup at risk Dernière sauvegarde trop ancienne sur DECIPROD : 80h, au-delà du seuil 54h pour une cadence quotidienne détectée.	Critical	Active	07/11/2026, 08:59 PM	—	Ongoing
	Low health score Score de santé					

 **An honesty worth highlighting.** If an administrator **disables** a database's supervision while an alert is open, Deynao does **not** mark it "Resolved" (which would suggest the problem is fixed). It closes it as **"Supervision stopped"** — a distinct status. *Disabling is not resolving*: the tool never lies to you about the real state.

Step 8 — The send log: the audit of notifications

Every e-mail Deynao sends — or **tries** to send — is **traced**. The **Send log** (reserved for the administrator) is the **complete audit** of notifications.

Send log ⓘ How does the Send log work?

All notification e-mails sent by Deynao (audit) — filter, paginate, export.

All Sent Failed Export to Excel Refresh

Search (database, type, message)...

Type: All types PERIOD: 24h 7d 30d **All** Per page: 50 Reset

57 sent 1 failed

DATE	TYPE	STATUS	RECIPIENTS	DETAIL	SUBJECT
07/11/2026, 07:30 AM	Daily report	Sent	2 recipient(s)	—	[Deynao] Rapport quotidien — 2 alerte(s) critique(s...
07/10/2026, 07:30 AM	Daily report	Sent	2 recipient(s)	—	[Deynao] Rapport quotidien — Tout va bien - 25 ba...
07/09/2026, 07:30 AM	Daily report	Sent	2 recipient(s)	—	[Deynao] Rapport quotidien — Tout va bien - 25 ba...
07/08/2026, 07:30 AM	Daily report	Sent	2 recipient(s)	—	[Deynao] Rapport quotidien — Tout va bien - 25 ba...
07/07/2026, 07:30 AM	Daily report	Sent	2 recipient(s)	—	[Deynao] Rapport quotidien — 2 alerte(s) critique(s...
07/06/2026, 07:51 AM	Daily report	Sent	2 recipient(s)	—	[Deynao] Rapport quotidien — 1 alerte(s) critique(s...
07/06/2026, 07:30 AM	Daily report	Failed	2 recipient(s)	SMTP server unreachable (connection refused)	[Deynao] Rapport quotidien — 1 alerte(s) critique(s...
07/05/2026, 07:30 AM	Daily report	Sent	2 recipient(s)	—	[Deynao] Rapport quotidien — Tout va bien - 25 ba...

- Filterable **All / Sent / Failed**, by period, and **exportable** (Excel/CSV).
- Each row: date, type, **status**, **number** of recipients, subject, and — on failure — a **plain-language error** (e.g. "SMTP server unreachable (connection refused)").

A "How does the Send log work?" button explains its logic:

How does the Send log work? ⓘ

Proof that Deynao notified you — or not

The log records EVERY attempt to send a notification e-mail (alert raised, reminder, resolution, daily report): the durable proof that Deynao tried to notify you — and whether the message went out or not.

Sent / Failed

- 🟢 "Sent": the e-mail was handed off to your mail server.
- 🔴 "Failed": the e-mail could not be delivered. The "Detail" column explains why in plain language (server unreachable, authentication refused...).

Don't confuse it with the alert state

The log is about E-MAIL DELIVERY, not the alert state. An alert can be "Resolved" in the History (the problem is gone) while its resolution e-mail is "Failed" here (the mail server was down at that moment). The two views measure two different things — this is expected.

Retries ("attempt N")

For an alert that is STILL active whose send fails, Deynao retries with ever-increasing spacing (30s → 1min → 5min → 15min → 30min → 1h): retries slow down so as not to hammer a broken server or flood this log, and never give up (once SMTP recovers, the e-mail goes out within ≤ 1h). "attempt 2", "attempt 3"... indicate these retries. A one-off notification (resolution, daily report) is not retried.

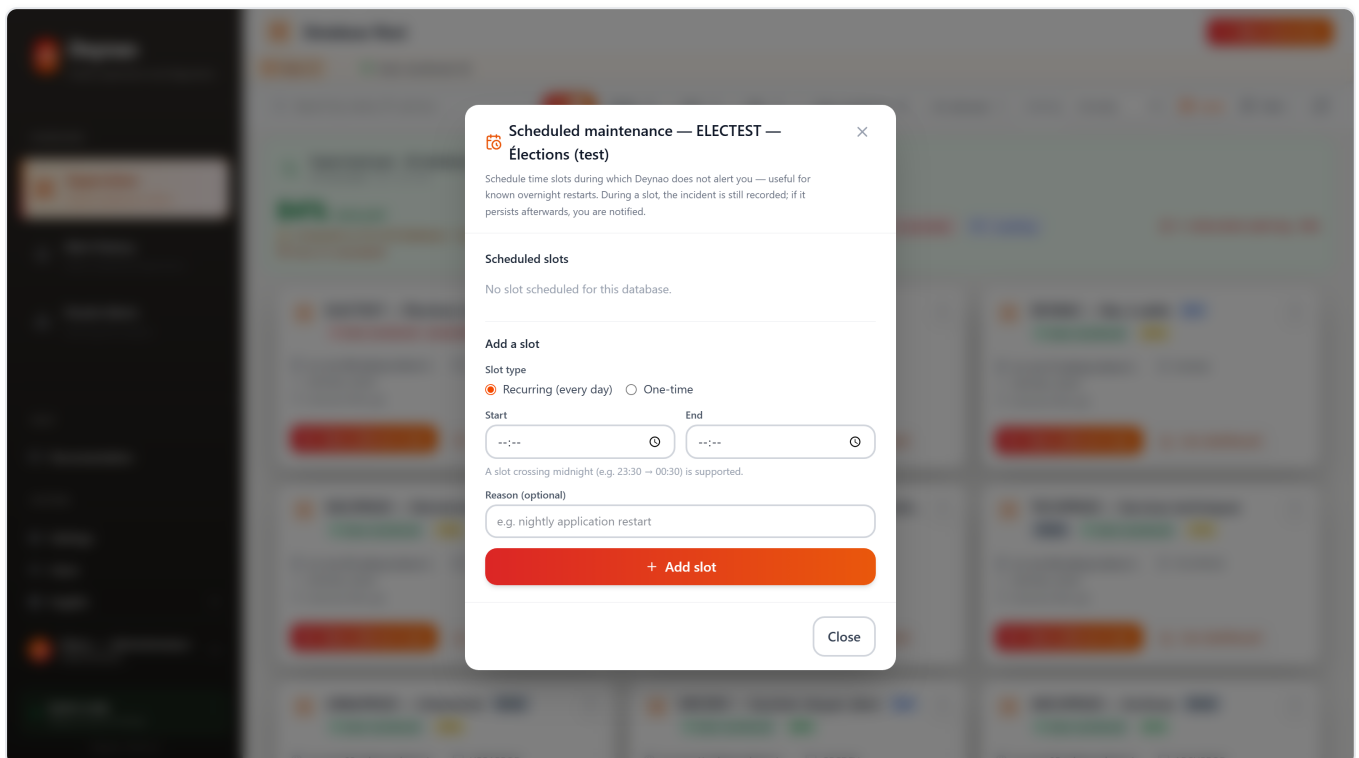
Close

 **The smart back-off.** If the SMTP server goes down, Deynao **does not flood** your inbox and **does not give up**: it **retries** with **increasing** delays — 30 s, 1 min, 5 min, 15 min, 30 min, then **once per hour** at most. As soon as SMTP is restored, the mail goes out.

 **A true, respectful audit register.** "*Delivery ≠ alert state*": a resolution e-mail that **could not go out** does **not** mean the alert is still open — the log distinguishes the two. Entries **purge themselves** (after 90 days), concern **only** Deynao (never your databases), and — a privacy detail — recipients' **addresses** are **never recorded**, only their **number**.

Step 9 — Maintenance windows

Your databases have **planned operations** (overnight restarts, batches) that would trigger **false alerts**. **Scheduled maintenance** (a database's : menu) makes Deynao **stay silent** during those windows.

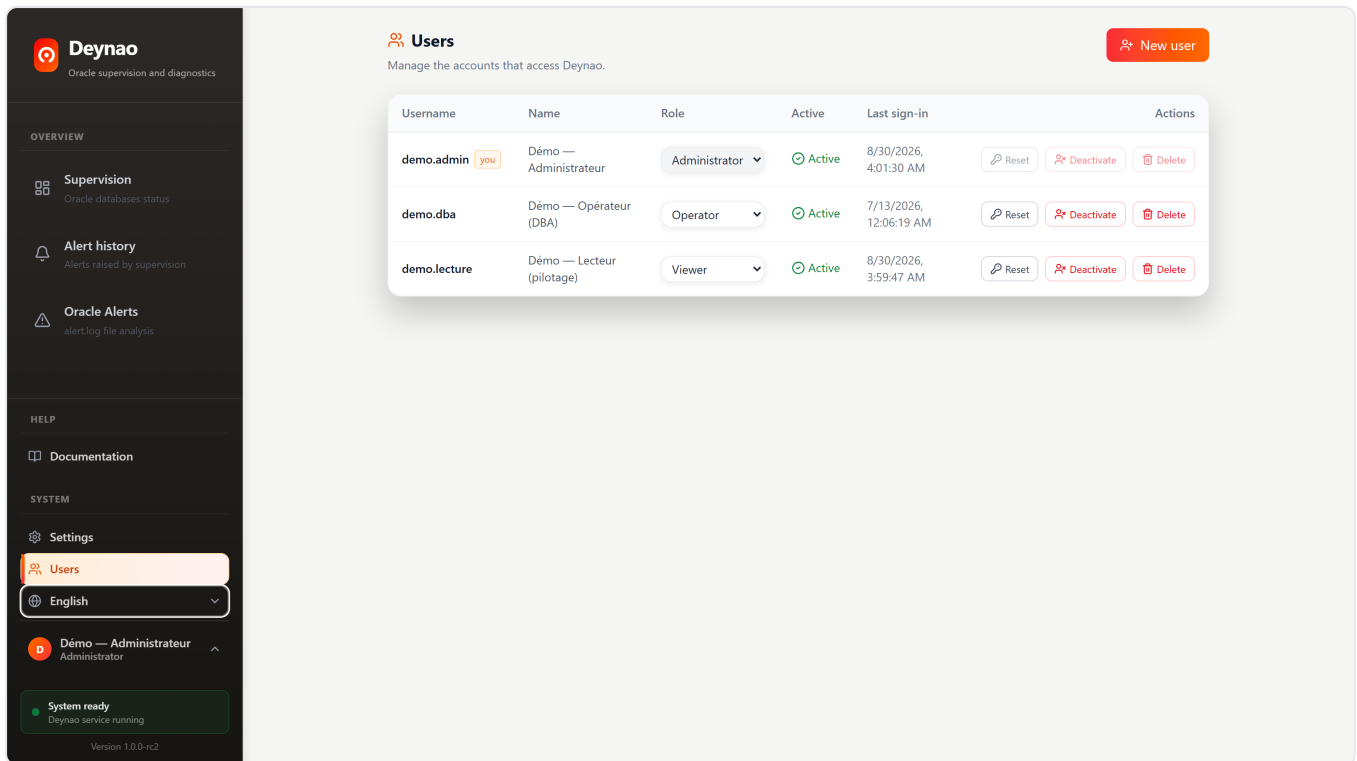


- **Recurring** (every day) or **One-time** — with a **start** and an **end**.
- A window that **crosses midnight** (e.g. 23:30 → 00:30) is correctly handled.
- An optional **reason** (e.g. "*nightly application restart*") documents the window.

 **Silent, but never blind.** During the window, Deynao **does not alert** — but the incident **is still recorded**. And **if it persists after** the window, you are **notified**. A badge on the database's card indicates it is in maintenance, and until what time.

Step 10 — Users and roles (RBAC)

Autonomous supervision acts **on your behalf**: it is therefore essential to know **who can do what**. Deynao distinguishes **three roles**, managed by the administrator in **Users**.

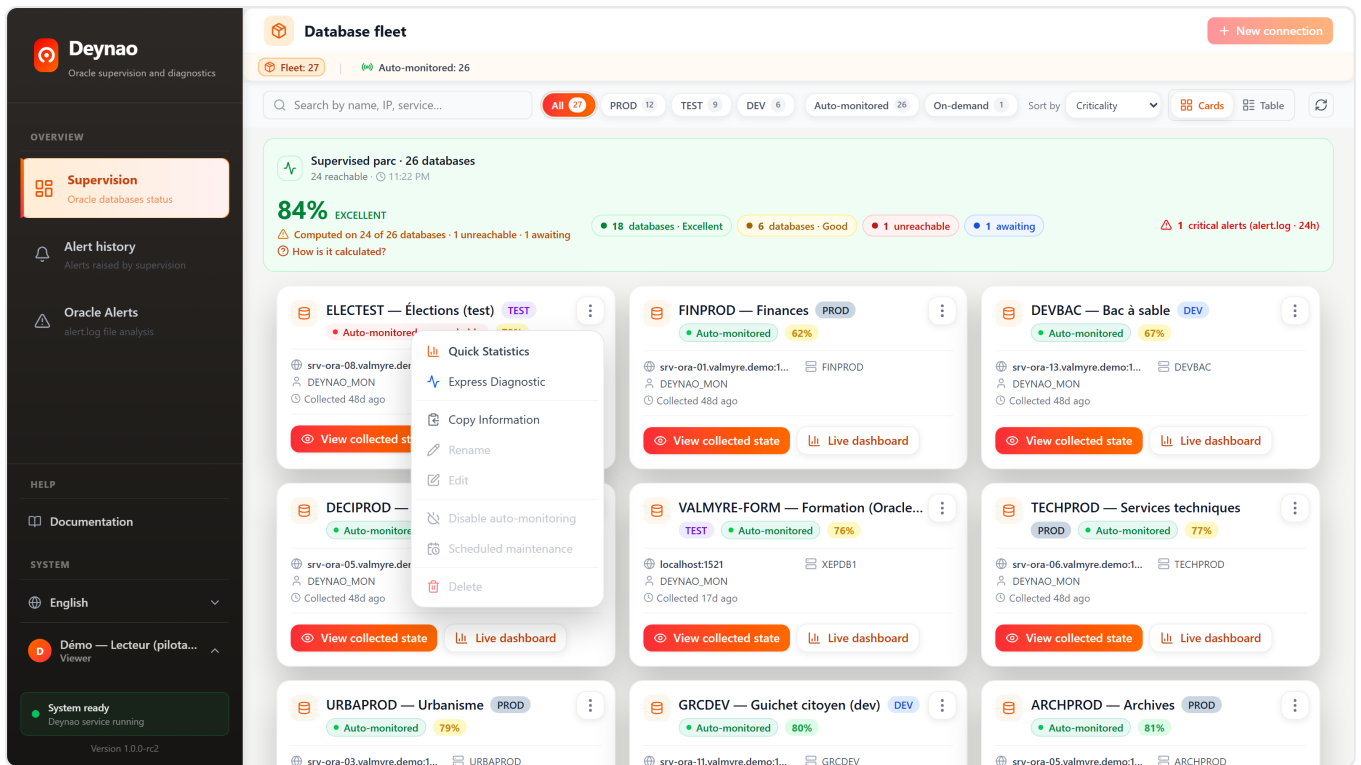


Username	Name	Role	Active	Last sign-in	Actions
demo.admin	Démo — Administrateur	Administrator	Active	8/30/2026, 4:01:30 AM	Reset Deactivate Delete
demo.dba	Démo — Opérateur (DBA)	Operator	Active	7/13/2026, 12:06:19 AM	Reset Deactivate Delete
demo.lecture	Démo — Lecteur (pilote)	Viewer	Active	8/30/2026, 3:59:47 AM	Reset Deactivate Delete

Role	Can do	Cannot
Administrator	Everything: users, settings (SMTP, licence), connections, supervision	—
Operator (DBA)	Supervise, connect databases, enable/disable monitoring, act	Manage users
Viewer	View everything (fleet, dashboards, history, reports)	Change anything: neither enable monitoring, nor manage accounts

The administrator **creates** accounts ("New user"), **resets** a password, **deactivates** or **deletes** an account, and sees everyone's **last sign-in**.

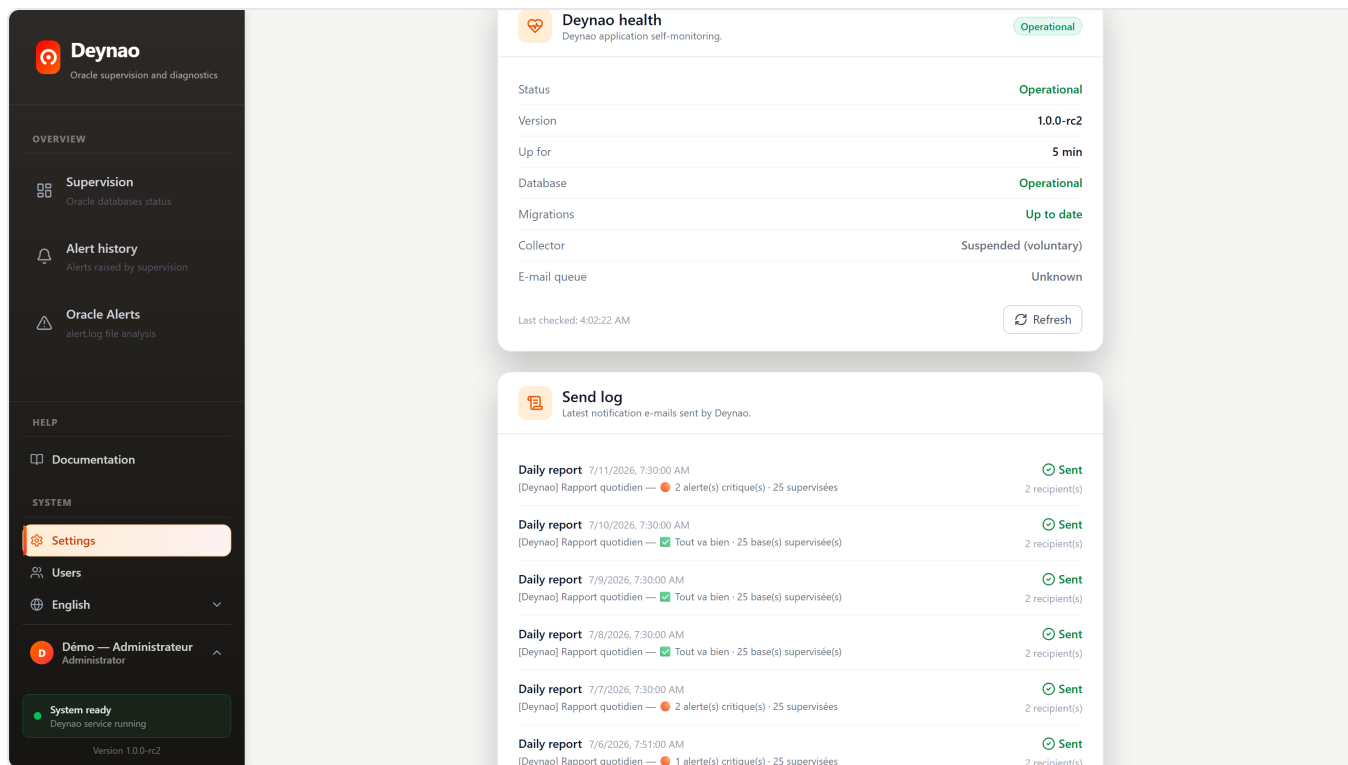
A concrete example — the Viewer. A *Viewer* account sees everything but touches nothing. In a database's menu, its **modifying** actions (Rename, Edit, **Disable auto-monitoring**, Scheduled maintenance, Delete) are **greyed out**: only the **read** actions remain (Quick Statistics, Express Diagnostics, Copy). And the **Settings** and **Users** menus are **inaccessible** to it.



👤 The right role for the right person. A decision-maker or a manager gets a **Viewer** access: they consult the fleet's state and the reports, **with no risk** of changing a configuration or interrupting a supervision.

Step 11 — The safety of autonomous supervision

Supervising **continuously** must **never** weaken your databases — nor the tool itself. Deynao goes as far as **monitoring itself**.



The **Deynao Health** card (in Settings) gives the application's **self-diagnosis**: overall state, **collector**, internal database, **migrations**, **e-mail queue**. A **safeguard** to know, at a glance, that supervision is **alive and well** — complemented by the "sign of life" from the **daily report** (Step 5.4).

And the underlying guarantees, recalled:

Guarantee	What it protects
Anti-lockout	Never a locked Oracle account (authentication circuit-breaker)
Network back-off	Never a hammered listener (increasing delays, up to once/h)
Read-only	Never a modification of your databases
Zero pollution	Never a line written into your <code>alert.log</code>
Never give up	A failed e-mail is retried, never lost; an alert stays open as long as the problem persists

 **The cardinal principle, applied continuously.** A supervision running 24/7 is all the more bound never to put the supervised database at risk. That is the promise Deynao keeps on **every** collection, **every** connection, **every** alert.

You are ready

Your fleet is now supervised **continuously and safely**: a **dedicated** read-only account, a non-intrusive **collector**, **transparent alert rules**, **audited** and retried **email alerts**, a **daily report** that also serves as a sign of life, **maintenance windows** for planned operations, and clear **roles** so that each person acts only within their scope.

What's next? All that remains is to **deploy** Deynao durably on your infrastructure (workstation or server, as a Windows service or a container). That is the subject of the next guide: "*Installing Deynao*".