FOR IT AND SECURITY TEAMS

DbVisualizer for Enterprise IT

DbVisualizer can be rolled out, locked down and standardised across an organisation from one central configuration file in the installation. This overview is organised by what IT and security teams typically want to achieve.

COMING marks capabilities in development; timing may change. Grey lines give the setting names, files and User Guide pages for readers who want the details.

WHERE CONFIGURATION LIVES

`DBVIS-HOME/resources/dbvis-custom.prefs`	Organisation-wide policy and defaults as `key=value` lines. Some settings are enforced; others are defaults users can change. Restart after editing. Special Properties
`DBVIS-HOME/dbvis.vmoptions`	Memory and Java options for the installation (macOS: `DbVisualizer.app/Contents/vmoptions.txt`); each user can add `~/.dbvis/user.vmoptions`. Installation Structure
`DBVIS-HOME/resources/profiles/`	Custom database profiles for every user.

At a glance

YOU WANT TO…	AVAILABLE NOW	COMING
Roll out to many users	Silent installs, per-machine install, central JVM and memory settings	–
Manage licences and access	Per-user licence keys or one key for the whole organisation	Required login; automatic licence delivery; add and revoke users in the customer portal; usage per user
Control what leaves the machine	Turn off usage statistics, error reporting, update checks, AI and online sign-in centrally	–
Govern AI use	Turn the AI Assistant off for everyone; approval rules for AI-generated SQL	Use your own or an in-house AI provider
Protect credentials	Require a master password with your complexity rule; encrypted credential storage	Unlock with macOS Keychain, Touch ID or Windows Hello
Work in restricted networks	Proxy support, your own certificate store, internal driver repositories, pre-packaged drivers	–
Audit and add guardrails	Central audit log; per-environment approval rules; turn off data editing	–
Customise database support	Database profiles: tailor navigation, views and actions, or add support for an in-house database; deploy to every user	–

YOU WANT TO…	AVAILABLE NOW	COMING
Standardise team setup	Shared connection folders on a network drive or in Git; shared scripts; settings packages	–
Use your identity provider	Database sign-in with Entra ID, AWS IAM, Okta, Kerberos and more	–
Automate	Command-line tool for scripts and scheduled jobs	Passwords from files or environment variables

Roll out to many users

DbVisualizer installs silently with a response file, so it fits into existing software distribution.

- **Installers for every platform:** Windows (installer with bundled Java, or zip), macOS (disk image for Apple silicon and Intel), Linux (RPM, DEB, tar.gz or installer).
- **Silent, scripted installs** with a response file that records the chosen options. Install for all users of a machine or for the current user only.

  Run the installer with `-q`, optionally `-dir <path>` and `-varfile <file>`; a manual install leaves a template in `DBVIS-HOME/.install4j/response.varfile`. Installing
- **One configuration file per installation** sets organisation-wide policy and defaults for every user of that install.

  `DBVIS-HOME/resources/dbvis-custom.prefs`. Special Properties
- **Central JVM settings** for memory and Java system properties, with an optional per-user addition.

  `dbvis.vmoptions`, for example `-Xmx8g`; per user `~/.dbvis/user.vmoptions`.
- **Java runtime included.** The bundled runtime is recommended; you can point at your own if policy requires it.

  `DBVIS_JAVA_HOME` for the installer, `JAVA_EXEC` for the launch scripts.
- **Seed user settings** by exporting a settings package (connections, drivers, preferences) and importing it, or by placing settings files in each user's workspace.

  File → Export Settings / Import Settings; start with `-workspace <folder>` to use another settings folder than `~/.dbvis`. Exporting and Importing Settings, Working with Workspaces

Manage licences and user access

Today you choose between a licence key per user and a single key for the organisation. Soon, licences can be delivered and managed through the DbVisualizer customer portal.

AVAILABLE NOW

- **Per-user licence keys**, installed by each user or distributed as files.

  Help → License Key, or copy `dbvis.license` to `~/.dbvis/`. Installing a Pro Edition License
- **One organisation-wide key**, placed once in the installation so every user is licensed.

  `DBVIS-HOME/resources/dbvis.license`
- **Offline verification.** Licences are signed files checked locally, with no activation step.

COMING

- **Required login.** Users sign in when DbVisualizer starts. You choose whether signing in is mandatory, or whether users who skip it fall back to the free edition.

  Set in `dbvis-custom.prefs`.

- **Automatic licence delivery.** A subscription key in your installation links users to your organisation, so a licence is created the first time a new user signs in.
  Set in `dbvis-custom.prefs`.
- **User management in the customer portal.** Add and revoke licences per user, and see usage per user.
- **Grace period** for users who are temporarily offline, with advance warnings.

Control what leaves the machine

The services that send data to DbVisualizer can be switched off for all users from the central configuration file. For telemetry and AI, users see the setting marked "Controlled by administrator".

- **Usage statistics and error reporting:** force on or off for everyone. Queries, data and anything identifying are never collected.
  `dbvis.-Telemetry=off` (or `on`). Telemetry
- **Update checks:** turn off completely, so updates are only installed through your own distribution.
  `dbvis.url.clientapi.checkforupdate=` (empty value); driver update checks: `dbvis.version.check.interval=-1`. Checking for Updates
- **AI Assistant:** turn off and hide for everyone.
  `dbvis.-AiDisabled=true`
- **Online sign-in and licence checks:** turn off, so the app makes no calls to DbVisualizer's account and licence services.
  `dbvis.-LoginDisabled=true` (Pro licences; also turns off AI)
- **Driver downloads:** point users at your own repository instead of public ones (see restricted networks).

> Unless the AI Assistant is used, queries and result data go only to your databases. A list of the external services the app can contact is available on request for firewall reviews.

Govern AI use

The AI Assistant is opt-in, and administrators can turn it off for the whole organisation.

AVAILABLE NOW

- **Turn AI off for everyone** from the central configuration file.
  `dbvis.-AiDisabled=true`
- **Approval rules for AI-generated SQL**, separate from and stricter than the rules for SQL users type themselves. They are restrictive by default.
  Tool Properties → AI Assistant → Permissions. Using AI — Permissions
- **Control what context is sent**, such as database structure, and choose the model.
  Using AI

COMING

- **Bring your own AI provider.** Connect DbVisualizer to an AI service your organisation already approves, including an in-house or self-hosted model, defined centrally by the administrator.

Protect credentials and meet security policy

Saved database, SSH and proxy passwords are always stored encrypted, and administrators can require a master password that meets their own complexity rule.

AVAILABLE NOW

- **Require a master password** for every user, and define its complexity rule and the message users see.
  Require: `dbvis.-MPR=true`. Rule: `dbvis.-MasterPasswordRule=<regex>` and `dbvis.-MasterPasswordRuleDescr=<message>`. Setting a Master Password
- **Choose whether passwords are saved** per connection, or ask users every time.
  The connection's Properties tab → Authentication.
- **Password-protect shared connection folders**, with your own complexity rule, or stop sensitive values from being shared at all.
  `dbvis.-MountedConnectionPasswordRule` and `-MountedConnectionPasswordRuleDescription`; `dbvis.sharedSecrets.enabled=false`
- **Use your own certificate store** for TLS, managed centrally, including client certificates.
  `dbvis.cacerts.path` and `dbvis.cacerts.password`, or `-Djavax.net.ssl.trustStore` / `keyStore` in `dbvis.vmoptions`. SSL
- **SSH tunnelling** with password, public key or SSH agent authentication, OpenSSH config files, known-hosts checking and jump hosts.
  Using SSH

COMING

- **Unlock with the operating system:** macOS Keychain with Touch ID, and Windows Hello.

Work in restricted or air-gapped networks

DbVisualizer runs without internet access once drivers are in place, and can get drivers from your own repositories.

- **Internal driver repositories.** Use your own Maven repository (such as Nexus or Artifactory) or a shared folder instead of public ones.
  Tool Properties → Driver Manager: remove the default repositories, add yours (a `file://` URL works). Set per user; distribute in a settings package. Maven Repositories
- **Pre-packaged drivers.** Download drivers once and distribute them to users with no network access.
  Download in a clean workspace, then copy its `drivers` folder to each user's workspace. Offline use
- **Proxy support**, using system settings or a configured proxy, including proxies that require sign-in.
  Tool Properties → General → Proxy Settings; defaults can be set in `dbvis-custom.prefs`.
- **Corporate certificates.** Trust your organisation's certificate authority through a central certificate store.
  See your own certificate store.
- **No activation.** Licences are verified offline.

Audit activity and add guardrails

An audit log records what each user runs, and approval rules add a safety net for production databases.

- **Audit log, controlled centrally.** Records executed SQL and, optionally, metadata calls and connection changes, as one JSON line per event. Passwords and tokens are masked. Administrators choose what is logged, where files go and how long they are kept, and users cannot turn it off.
  `dbvis.auditLog.enabled=true`, `.logMetadata`, `.logConnectionChanges`; `dbvis.-auditLog.directory`, `.retentionDays`, `.maxFileSizeBytes`. Audit Logging

- **Approval rules per environment.** Mark connections as Development, Test or Production, then allow, deny or ask for confirmation per type of SQL statement, data edit or connection.
  Tool Properties → General → Permissions. Permissions in the SQL Commander, Permissions for Table Data Editing
- **Turn off data editing** in result grids for all users.
  `dbvis.disabledataedit=true`. Read-Only Connections

These guardrails work alongside database permissions, which remain the authority on what a user may change.

Customise database support with profiles

Database profiles are one of DbVisualizer's most powerful customisation routes. They define what users see and can do for each database, and organisations can extend them or write their own.

`Navigation` Which objects appear in the database tree and how they are grouped.	`Views` The tabs that show details for each object, driven by your own SQL.
`Actions` Right-click operations with input forms, confirmation and generated SQL, for example wrapping internal administration procedures in a guided dialog.	`On connect` Commands run when a connection opens, such as setting session options.

WHAT ORGANISATIONS DO WITH THEM

- **Tailor a supported database** to how your company works: show each user's own schema first, hide object types nobody uses, add views for in-house metadata or monitoring tables.
- **Add support for a database DbVisualizer does not cover**, such as an in-house or niche system, on top of its JDBC driver.
- **Change only what you need.** A custom profile extends the built-in one, so it keeps receiving DbVisualizer's improvements.

Profiles can be deployed in the installation for every user, or kept in a shared folder or Git repository so the team works from one version.

XML files in `DBVIS-HOME/resources/profiles/`, or a folder added under Tool Properties → General → Database Connection → Database Profile. Database Profiles, Extending, Creating, How a profile is loaded

Standardise team setup

Teams can share one set of connections and scripts, kept on a network drive or in a Git repository.

- **Shared connection folders.** Keep connection definitions in one place; everyone who mounts the folder gets the same connections and receives updates. Each user's credentials stay private, unless you choose to share them in a password-protected folder.
  Databases tab → Mount Database Connection Folder. Sharing Database Connections

- **Built-in Git support.** Clone repositories, review changes and publish them from inside DbVisualizer, for both connections and SQL scripts.
  Git tab. Working with Git
- **Shared script folders** for common SQL, on a network drive or in Git.
  Files tab → mount folder. Using Files
- **Settings packages** to give new users a ready-made setup.
  See Roll out to many users.

Use your identity provider for database sign-in

DbVisualizer supports the sign-in methods offered by each database's driver, so users can connect with their corporate identity instead of database passwords.

Set in the connection's Driver Properties. Single Sign-On, 2FA and MFA

DATABASE OR PLATFORM	SIGN-IN OPTIONS
Azure SQL	Microsoft Entra ID: interactive, integrated, password, managed identity, service principal
SQL Server	Windows integrated authentication
Amazon Athena	AWS IAM, IAM Identity Center, Okta, Ping, AD FS, Azure AD, SAML, JWT
Amazon Redshift, DynamoDB	AWS IAM
Databricks	OAuth, machine-to-machine, Azure managed identity, access tokens
Google BigQuery	Service account, user OAuth, application default credentials
Other databases	Kerberos and other methods supported by the JDBC driver

Automate with the command line

A command-line tool runs SQL and scripts without the user interface, for scheduled jobs and CI pipelines.

`dbviscmd`. Command Line Interface

AVAILABLE NOW

- Run SQL statements or script files against saved or ad-hoc connections, with variables passed in from shell scripts.
  `-connection`, `-url`, `-sql`, `-sqlfile`
- Create connections from the command line, for provisioning.
  `-createconnection`; `-masterpw` for encrypted passwords
- Clear exit codes and logging for unattended jobs.

COMING

- Read passwords from a file, an environment variable or standard input, so they never appear on the command line.