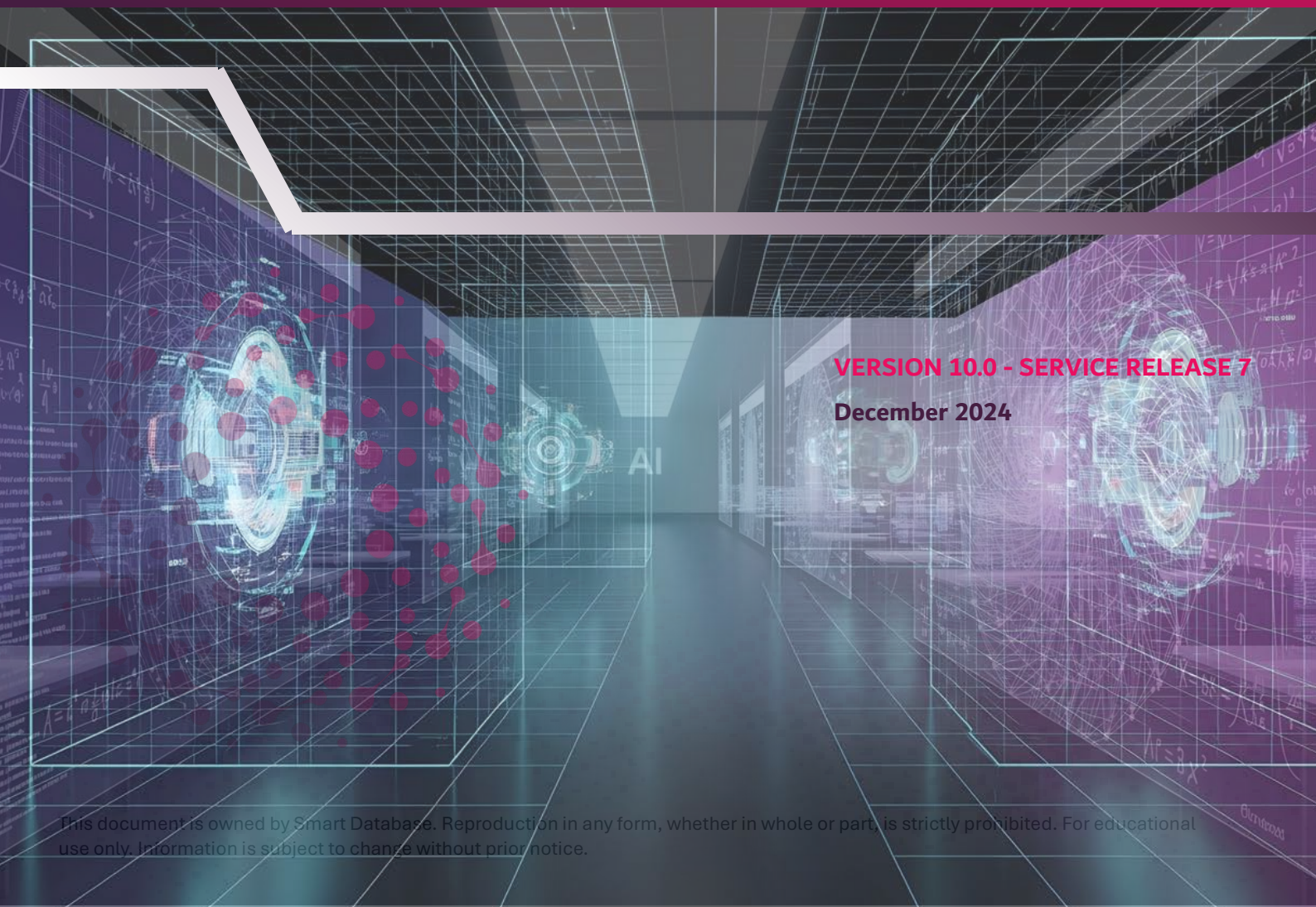
Smart Database

Knowledge Center

Participant Guide –

Application Configuration



VERSION 10.0 - SERVICE RELEASE 7

December 2024

Smart Database

About This Document

This document applies to SmartDB (Smart Database) Version 10.0 and all subsequent releases. The specifications and information contained herein are subject to change without prior notice. Any changes will be documented in future release notes or updated editions of this guide.

Copyright Notice

License and Usage Terms

Use of this software is governed by the SmartDB (Smart Database) License Agreement and applicable commercial terms provided at the time of purchase or deployment.

The license terms are included as part of the SmartDB (Smart Database) product documentation and/or are available in the root directory of the installed SmartDB (Smart Database) software. Unauthorized reproduction, distribution, or modification of this software or documentation is strictly prohibited.

Third-Party Software

SmartDB (Smart Database) may include or integrate with third-party software components, open-source libraries, or external services. Applicable third-party copyright notices, license terms, disclaimers, and additional rights or restrictions are provided in the Third-Party Licenses documentation included with the product distribution and/or located in the installation directory. Use of such third-party components is subject to their respective license terms.

Trademarks

The name SmartDB , Smart Database, and all SmartDB-related product names, logos, and services are either trademarks or registered trademarks of SmartDB inc. and/or its affiliates.
All other company names, product names, and trademarks mentioned in this document are the property of their respective owners and are used for identification purposes only.

Disclaimer

This documentation is provided “as is” without warranty of any kind, either express or implied. SmartDB.ai / Script Technical Solutions LLC shall not be liable for any direct, indirect, incidental, or consequential damages arising from the use of this document or the software described herein.

Smart Database

Contents

1. Introduction

1.1 Scope of Document

This document defines the scope of application configuration activities for SmartDB (Smart Database) and serves as a comprehensive guide for configuring the core functional components of the platform required to enable secure data access, intelligent analytics, and AI-driven capabilities. It outlines the configuration responsibilities, participation requirements, and technical considerations necessary to operationalize SmartDB within an organizational environment. The application configuration covered in this document focuses on enabling SmartDB to connect with enterprise data sources, manage unstructured content, integrate advanced AI and language models, enforce identity and access controls, and establish semantic intelligence through knowledge modeling. To ensure clarity, consistency, and effective implementation, the application configuration is structured into a set of well-defined configuration streams, each addressing a specific functional area of the SmartDB platform, as outlined below:

- **Structured Data Source Configuration (Databases)**
  Configuration of enterprise structured datasets, including secure connection to relational and analytical databases. This stream covers database connectivity setup, credential management, schema discovery, table and field mapping, data refresh policies, access control, and enablement of analytics, natural language querying, and predictive capabilities on structured data.
- **Large Language Model (LLM) Configuration**
  Configuration of the SmartDB LLM layer, including selection and integration of large language models (on-premise, private, or external), model routing, prompt governance, security and compliance controls, and performance tuning to support conversational analytics and advisory use cases.
- **Identity & Access Management Configuration (LDAP / Directory Services)**
  Configuration of user authentication and authorization through LDAP or directory services. This stream includes directory connection setup, user and group synchronization, role mapping, authentication policies, access segregation, and audit logging to ensure secure and compliant access to SmartDB services.

- **Unstructured Data Source Configuration (Document Libraries & Repositories)**
  Configuration of unstructured data sources such as document libraries and content repositories. This includes document ingestion rules, file format support, metadata extraction, indexing, classification, access permissions, and activation of AI-driven document advisory and search capabilities across connected repositories.
- **Knowledge Graph Configuration**
  Configuration of the SmartDB Knowledge Graph, including definition of entities, relationships, and semantic models. This stream covers knowledge extraction from structured and unstructured data, ontology configuration, graph enrichment, and integration with analytics, LLMs, and advisory modules to enable contextual reasoning and advanced insights.

This document is limited to application-level configuration and participation activities and explicitly excludes infrastructure provisioning, platform installation, and any custom development beyond the supported SmartDB configuration capabilities. It defines the scope of application configuration activities for SmartDB (Smart Database) and serves as a comprehensive guide for configuring the core functional components of the platform required to enable secure data access, intelligent analytics, and AI-driven capabilities, outlining the applicable configuration boundaries and responsibilities.

2. Scope of Configuration Activities

2.1 Structured Data Source Configuration (Databases)

Database connection configuration in SmartDB (Smart Database) is managed centrally through the platform's administrative interface. This configuration allows SmartDB to securely connect to structured data sources such as relational and analytical databases and enables downstream capabilities including data exploration, analytics, machine learning, and natural language querying.

To access the database connection configuration page, log in to the SmartDB platform using an account with administrative privileges. From the main dashboard, navigate to the user profile menu located in the upper-right corner of the screen. Open the profile menu and select JDBC Connections. This action directs the user to the Database Connections management interface, where all existing database connections are listed and new connections can be created or modified.

The JDBC Connections page serves as the centralized control point for managing structured data integrations within SmartDB. From this page, administrators can define new database connections, update existing configurations, validate connectivity, and control access to connected datasets based on user roles and permissions.

The screenshot below illustrates the navigation path and the location of the JDBC Connections option within the SmartDB user interface.

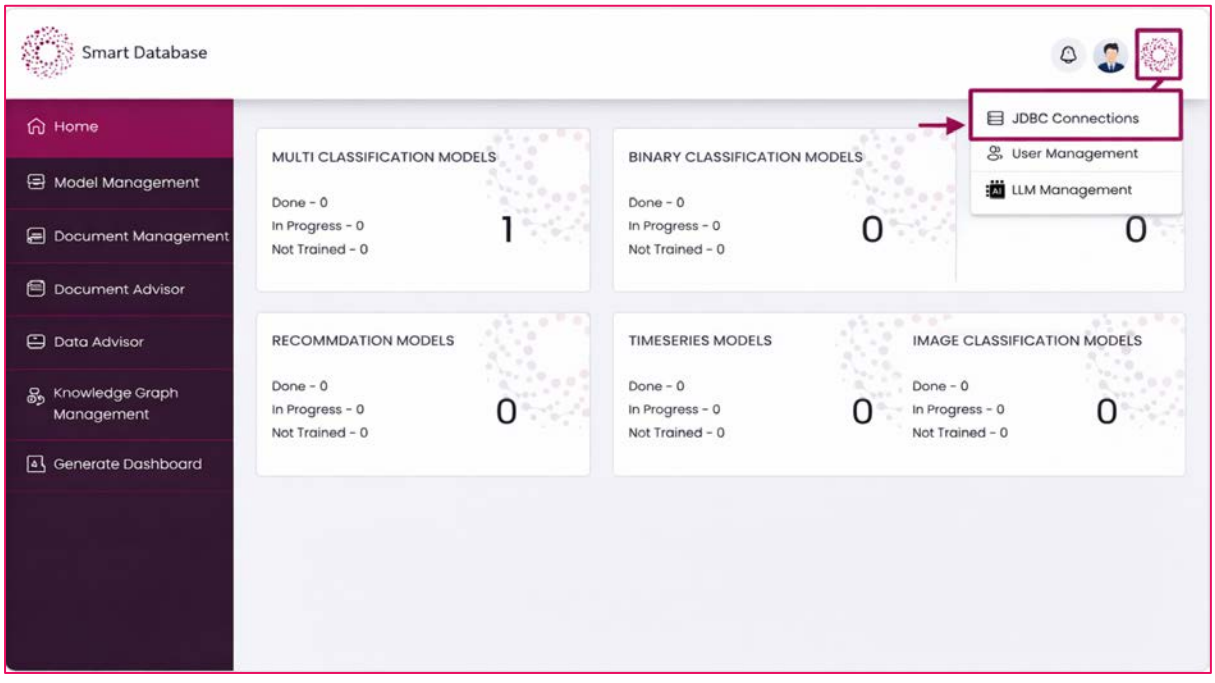


Figure 1: Accessing the JDBC Connections option from the main dashboard

After navigating to JDBC Connections from the main dashboard, the user is directed to the JDBC Connections Management page. This page provides a centralized view of all configured database connections within the SmartDB platform and serves as the primary workspace for managing structured data integrations as shown in **Figure2**.

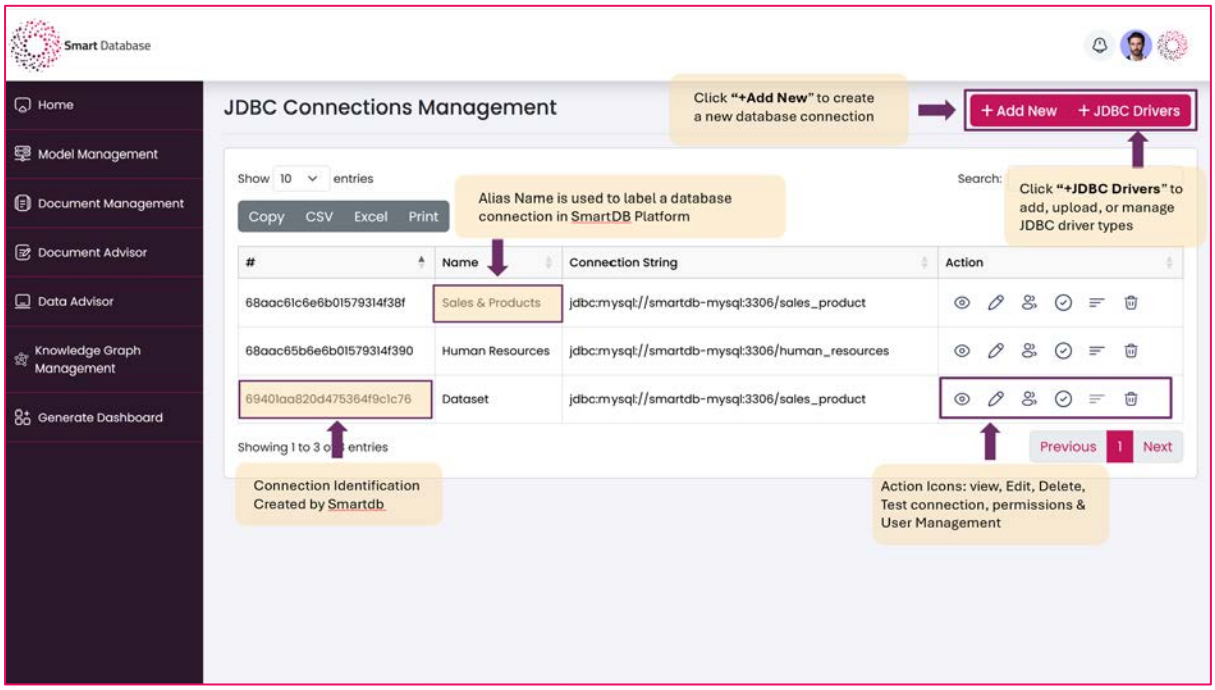


Figure 2: JDBC Connections Management page showing existing database connections and configuration options

The JDBC Connections Management page displays a tabular list of existing database connections, including the connection identifier, connection name, and JDBC connection string. From this page, administrators can easily review which databases are currently connected to SmartDB and monitor their configuration status. The table supports standard data management functions such as pagination, search, and export options (Copy, CSV, Excel, Print), enabling efficient administration and documentation of configured connections.

At the top-right corner of the page, two primary actions are available:

- **Add New**:
  This option allows administrators to create a new database connection. Selecting **Add New** opens the database connection configuration form, where users can define connection details such as database type, host, port, credentials, and connection parameters. Once configured, the new connection becomes available for use across SmartDB modules, including Data Advisory, AutoML, dashboards, and Knowledge Graph creation.

- **JDBC Drivers**:
  This option allows administrators to manage JDBC drivers required for database connectivity. From this section, users can upload, register, or manage JDBC driver files needed to support different database technologies. Adding the appropriate driver is a prerequisite for establishing new database connections that are not supported by default.

Each configured database connection includes an Actions column, providing quick access to common management operations such as viewing connection details, editing configuration settings, validating connectivity, managing access permissions, and removing connections when no longer required.

This page acts as the operational control center for structured data connectivity in SmartDB and represents the foundation for enabling analytics, machine learning, natural language querying, and semantic knowledge generation across connected enterprise databases.

JDBC Connection Configuration Options

SmartDB (Smart Database) supports two primary approaches for configuring database connections:

1. **Using Predefined JDBC Drivers (Recommended)**

SmartDB provides a set of predefined JDBC drivers for commonly used SQL and NoSQL databases, allowing users to quickly configure connections without additional setup. These predefined drivers cover widely adopted enterprise databases and are optimized for compatibility and stability.

When creating a new connection, users can select the appropriate database type from the predefined list and provide the required connection details, such as:

- Database alias name
- Host and port
- Database or service name
- Authentication credentials

This approach is recommended whenever the target database is already supported by SmartDB's built-in drivers.

2. **Using Custom JDBC Drivers (Advanced)**

If the required database technology is not available in the predefined driver list, SmartDB allows users to add custom JDBC drivers.

In such cases, users must first click "+ JDBC Drivers" from the JDBC Connections Management page to upload and register the required driver. Once the custom driver is added successfully, it becomes available for selection when creating a new JDBC connection.

This approach enables SmartDB to support additional or specialized database platforms while maintaining a unified configuration experience.

2.1.1 Add New Database connection (Recommended)

The Add New JDBC Connection function allows administrators to configure and register a new database connection within the SmartDB platform. Once a database connection is added, it becomes available for use across SmartDB modules, including Data Advisor, AutoML, Dashboards, Document Advisory, and Knowledge Graph.

The Add JDBC Connection page allows administrators to create and configure a new database connection in the SmartDB platform. This page is used to register both SQL and NoSQL databases, enabling SmartDB to securely access structured data for analytics, machine learning, data advisory, dashboards, and knowledge graph generation. Once a database connection is added, it becomes available for use across SmartDB modules.

SmartDB supports predefined JDBC drivers for commonly used databases and also allows the addition of custom JDBC drivers when required.

Purpose of Adding a JDBC Connection

Adding a JDBC connection enables SmartDB (Smart Database) to:

- Securely connect to structured enterprise datasets
- Discover database schemas, tables, and fields
- Enable analytics, machine learning, and natural language querying on the connected data
- Serve as a data source for Knowledge Graph generation and AI-driven insights

As shown in **Figure 3**, the Add JDBC Connection interface provides the required configuration fields for defining a new JDBC connection, including driver selection, connection details, and authentication parameters.

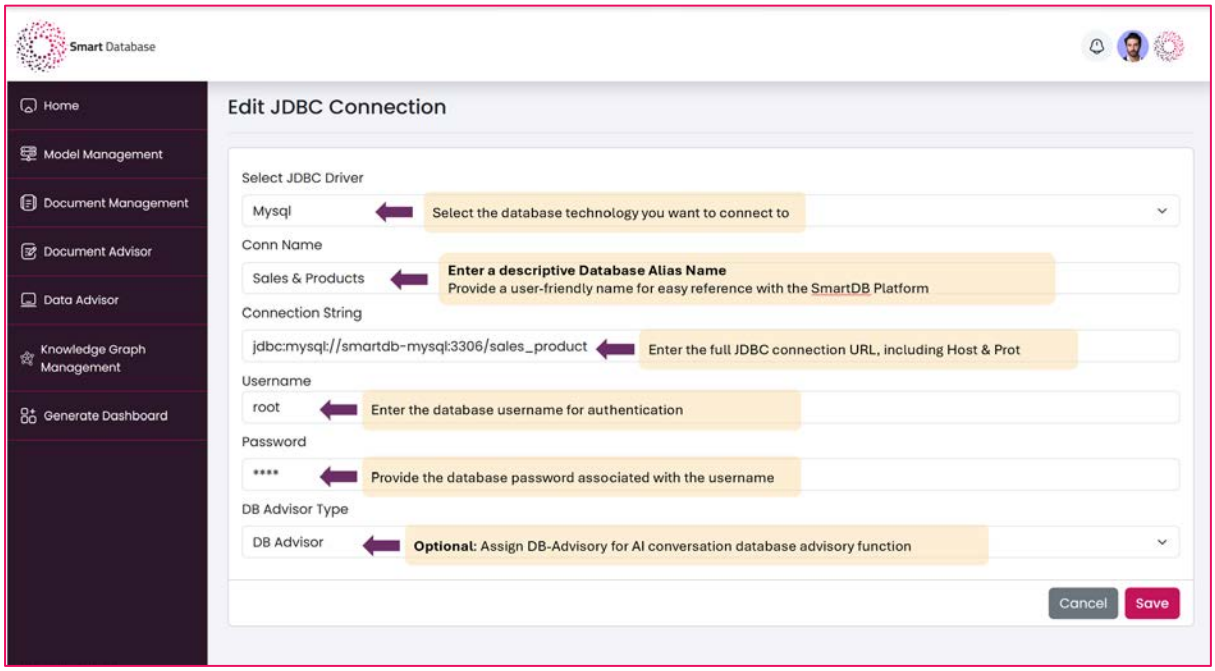


Figure 3: Add New JDBC Connection workflow within the SmartDB platform

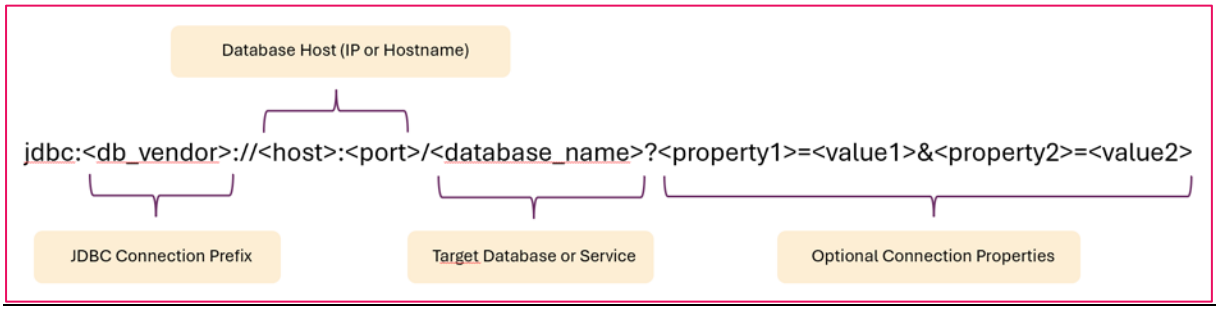
2.1.1.1 JDBC Connection String Construction

A JDBC connection string (JDBC URL) defines how the platform establishes a connection to a target database. It follows a standardized structure that begins with the jdbc prefix, identifying the connection as JDBC-based, followed by the database vendor identifier (such as MySQL, Oracle, PostgreSQL, or SQL Server).

The connection string then specifies the database server location, including the hostname or IP address and the listening port, along with the target database name, service name, or schema. Optional parameters can be appended to the connection string to control connection behavior, such as enabling encryption (SSL/TLS), setting connection and socket timeouts, defining character encoding, or handling time zone settings. Authentication credentials, including username and password, are typically provided separately to ensure better security practices.

A JDBC connection string is constructed using the following standard format:

```
jdbc:<db_vendor>://<host>:<port>/<database_name>?<property1>=<value1>&<property2>=<value2>
```



Below is the breakdown for the JDBC connection

Part	Meaning
jdbc	Indicates this is a JDBC connection
<db_vendor>	Database type / driver (mysql, postgresql, oracle, sqlserver, etc.)
<host>	Database server hostname or IP
<port>	Database listening port
<database_name>	Schema / SID / Service name
properties	Optional parameters (SSL, timeout, charset, etc.)

Below are commonly used JDBC connection string formats for supported databases. These examples illustrate how the standard JDBC structure is applied across different database vendors.

Database	JDBC Driver Prefix	Default Port	JDBC Connection String Example
MySQL	jdbc:mysql	3306	jdbc:mysql://<host>:3306/<database>?useSSL=false&serverTimezone=UTC
PostgreSQL	jdbc:postgresql	5432	jdbc:postgresql://<host>:5432/<database>
Oracle (Service Name)	jdbc:oracle:thin	1521	jdbc:oracle:thin:@//<host>:1521/<service_name>
Oracle (SID – Legacy)	jdbc:oracle:thin	1521	jdbc:oracle:thin:@<host>:1521:<SID>
Microsoft SQL Server	jdbc:sqlserver	1433	jdbc:sqlserver://<host>:1433;databaseName=<database>
MariaDB	jdbc:mariadb	3306	jdbc:mariadb://<host>:3306/<database>
IBM DB2	jdbc:db2	50000	jdbc:db2://<host>:50000/<database>

SAP HANA	jdbc:sap	30015	jdbc:sap://<host>:30015/?databaseName=<database>
SQLite	jdbc:sqlite	N/A	jdbc:sqlite:/path/to/database.db
Amazon Aurora (MySQL-Compatible)	jdbc:mysql	3306	jdbc:mysql://<endpoint>:3306/<database>
Amazon Aurora (PostgreSQL-Compatible)	jdbc:postgresql	5432	jdbc:postgresql://<endpoint>:5432/<database>

Below is an example of a valid JDBC connection string used to connect to a MySQL database:

- jdbc:mysql://smartdb-mysql:3306/sales_product

2.1.1.2 Configuration Fields Description

The Add JDBC Connection form consists of the following fields:

1. Select JDBC Driver

This field allows the user to select the database technology to connect to.

- SmartDB provides predefined JDBC drivers for popular SQL and NoSQL databases (such as MySQL, PostgreSQL, SQL Server, Oracle, MongoDB, etc.) as shown in **Figure 4**.
- Selecting a predefined driver automatically aligns the connection with the required driver configuration.
- If the required database type is not listed, users can add a new driver using the "+ JDBC Drivers" option before creating the connection.

Best Practice: Always use a predefined driver when available for better compatibility and support.

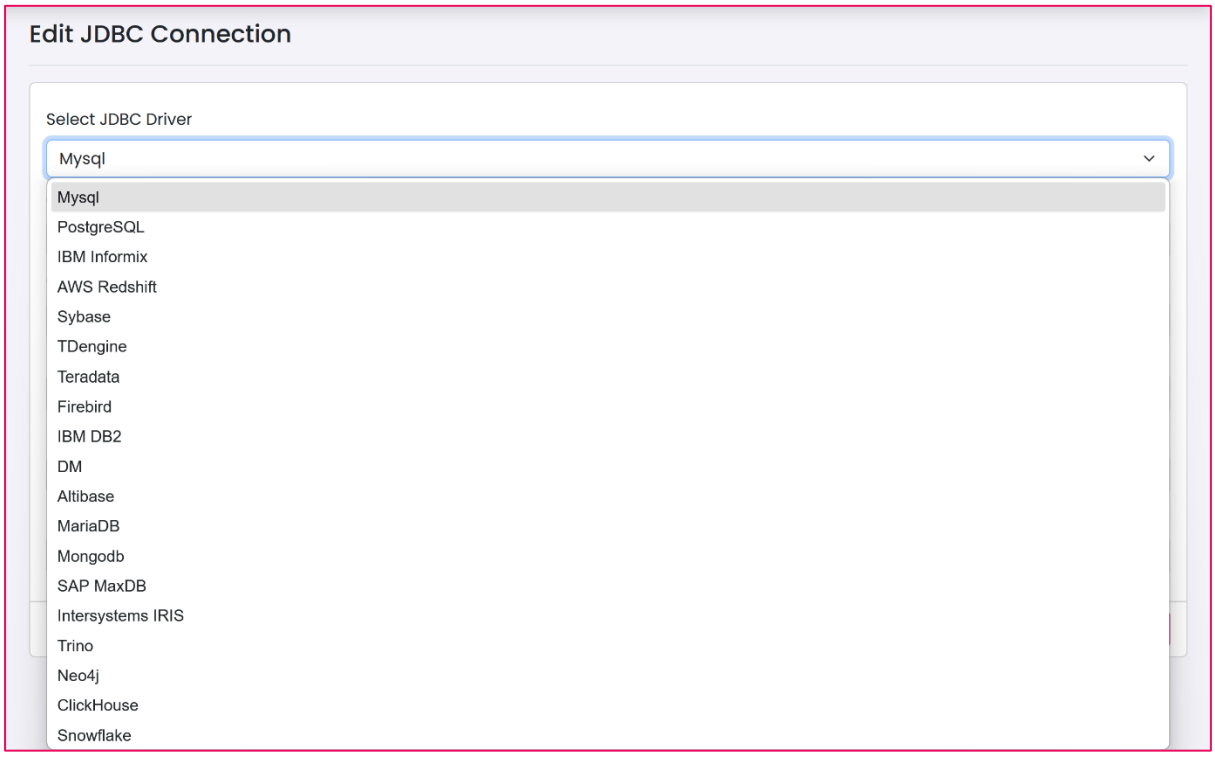


Figure 4: Out-of-the-box predefined JDBC drivers from SmartDB platform

2. Connection Name (Database Alias Name)

This field defines a logical alias name for the database connection.

- It is a user-friendly name used across the SmartDB platform.

- This name appears in Data Advisor, dashboards, AutoML workflows, and Knowledge Graph configuration.
- The alias name should be descriptive and reflect the purpose of the database (for example: *Sales Database*, *HR System*, *Finance Analytics*).

3. Connection String

This field requires the full JDBC connection string for the selected database.

- The connection string typically includes the database type, host, port, and database or service name.
- The format of the connection string depends on the selected JDBC driver.
- This field is mandatory and must follow the correct JDBC syntax for the selected database technology.

Example (MySQL): jdbc:mysql://hostname:3306/database_name

4. Username

This field specifies the database user account that SmartDB will use to connect to the database.

- The user must have sufficient permissions to read the required schemas, tables, or collections.
- It is recommended to use a read-only database user whenever possible to enhance security.

5. Password

This field contains the password associated with the specified database username.

- Credentials are securely stored and used only for establishing the database connection.
- Ensure the password is kept confidential and complies with organizational security policies.

6. DB Advisor Type

This optional field allows the user to select a DB Advisor Type, if applicable.

- The DB Advisor Type determines how SmartDB applies advisory, analytics, or optimization features for the connected database.
- If no advisory configuration is required, this field can be left as **None**.

Notice: This function used for activates the Database Advisory feature, allowing users to play with their database through natural language interaction, enabling conversational querying, guided exploration, and AI-driven insights directly on the connected data.

7. Saving the Connection

After completing all required fields:

1. Review the configuration details.
2. Click **Save** to validate and store the connection.
3. Once saved successfully, the new database connection will appear in the **JDBC Connections Management** list and will be available for use across SmartDB modules.
4. Click **Cancel** to discard the configuration and return to the previous page

2.1.1.3 Finalizing the Connection

Once all required fields are completed and the connection is successfully validated, the user can save the configuration. The newly created database connection will then appear in the JDBC Connections Management table and become immediately available for selection and use across SmartDB modules.

Figure 5 shows the successful database connection status, confirming that the database is properly connected and accessible by SmartDB services. A *Connected* status indicates that SmartDB can communicate with the database without errors. If a connection fails, users are advised to review the connection configuration and verify network and firewall settings with the system administrator.

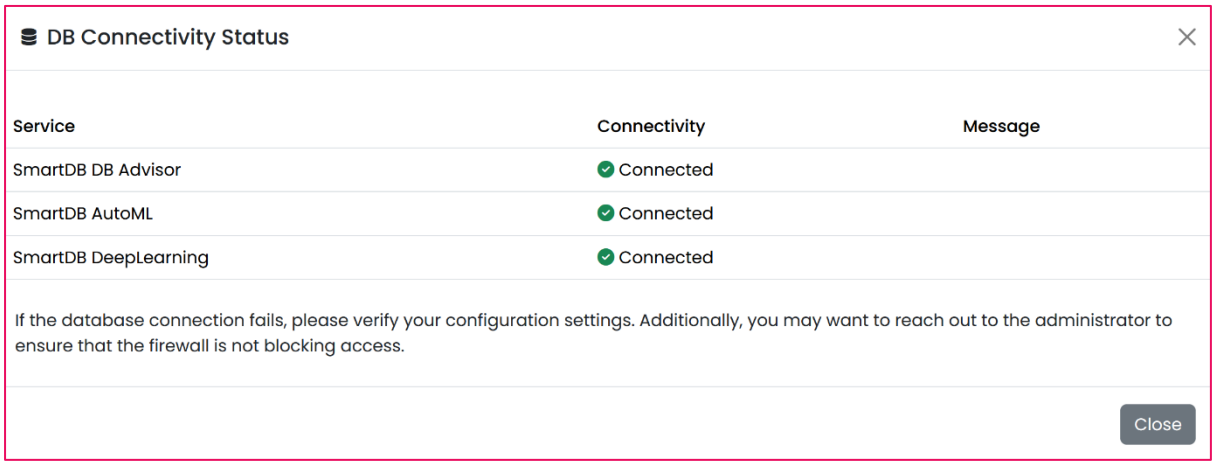


Figure 5: Successful database connection creation and connectivity validation in SmartDB

2.1.2 Add Custom JDBC Connection (Advanced)

SmartDB (Smart Database) provides a JDBC Drivers Management capability that allows system administrators to view, add, and manage JDBC drivers used for database connectivity. This feature gives full control to the administration team to extend SmartDB connectivity beyond the predefined database drivers shipped with the platform.

Through this capability, SmartDB can support custom, legacy, proprietary, or less common SQL and NoSQL databases, ensuring maximum flexibility and long-term scalability.

2.1.2.1 JDBC Drivers List Page

The JDBC Drivers page displays all JDBC drivers currently registered in the SmartDB platform.

From this page, administrators can:

- View all configured JDBC drivers
- Review driver names and associated driver class paths
- Add new JDBC drivers
- Remove existing drivers (if no longer required)

Each driver entry includes:

- **Driver Name** – A human-readable name identifying the database technology
- **Driver Class** – The fully qualified Java driver class used by JDBC
- **Actions** – Options to view details or delete the driver

This page acts as the central registry for all database drivers used across SmartDB platform as shown in **Figure 6**.

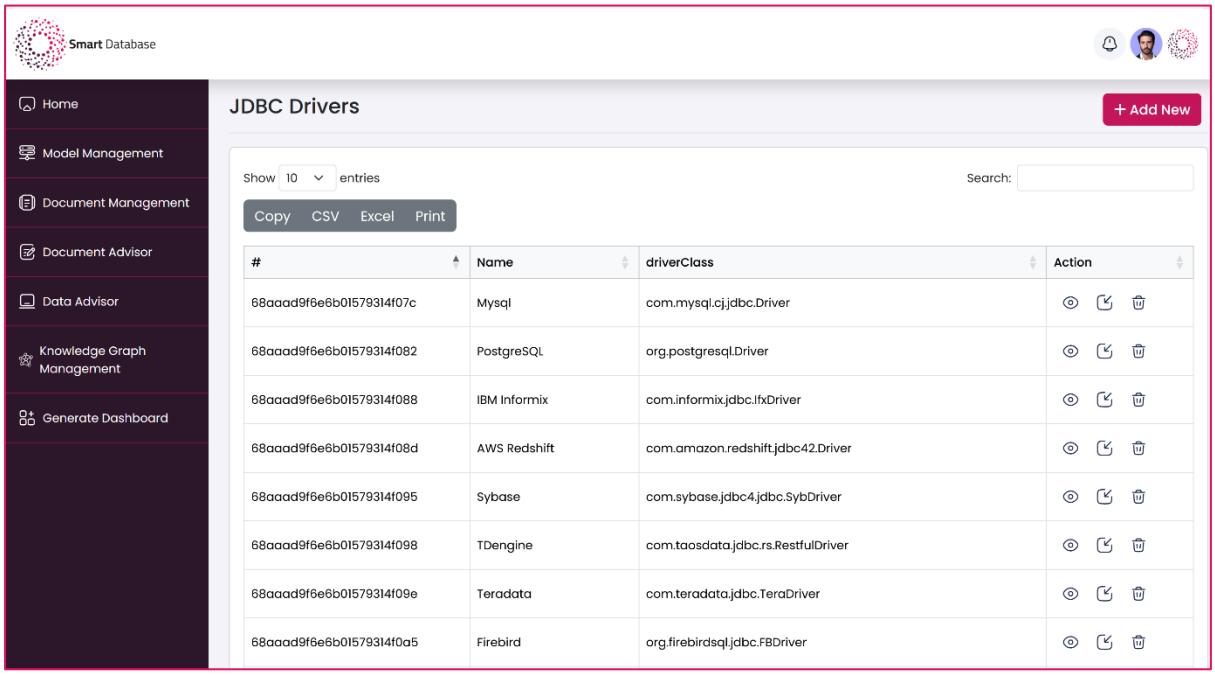


Figure 6: JDBC Drivers Management page

2.1.2.2 Add JDBC Driver Page

The Add JDBC Driver page allows administrators to register a new JDBC driver that is not predefined by SmartDB (Smart Database).

This functionality provides full administrative control and enables SmartDB to connect to **c**ustom or unsupported database systems.

2.1.2.2.1 Accessing the Add JDBC Driver Page

1. Navigate to JDBC Drivers from the administration menu.
2. Click “+ Add New”.
3. The Add JDBC Driver configuration form will be displayed.

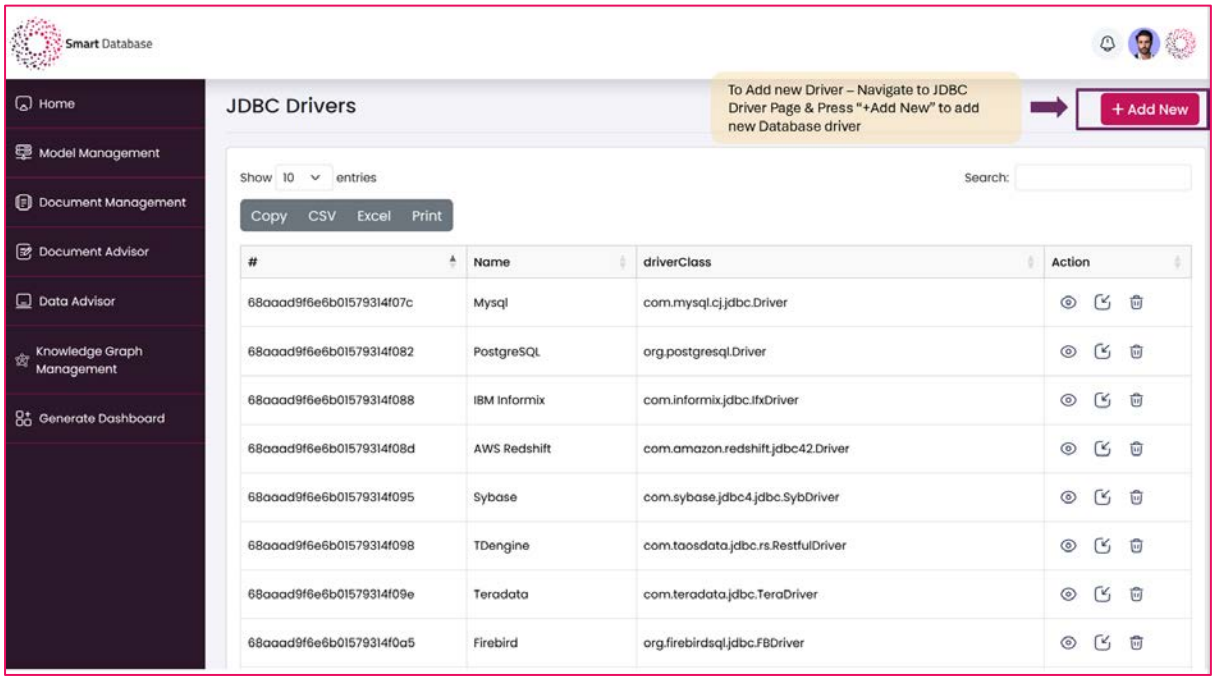


Figure 7: JDBC Drivers Page – Add new Database driver

2.1.2.2.2 JDBC Driver Configuration Fields

As shown in **Figure 8**, the Add JDBC Driver form consists of the following configuration fields, all of which must be completed accurately to ensure successful driver registration and usage.

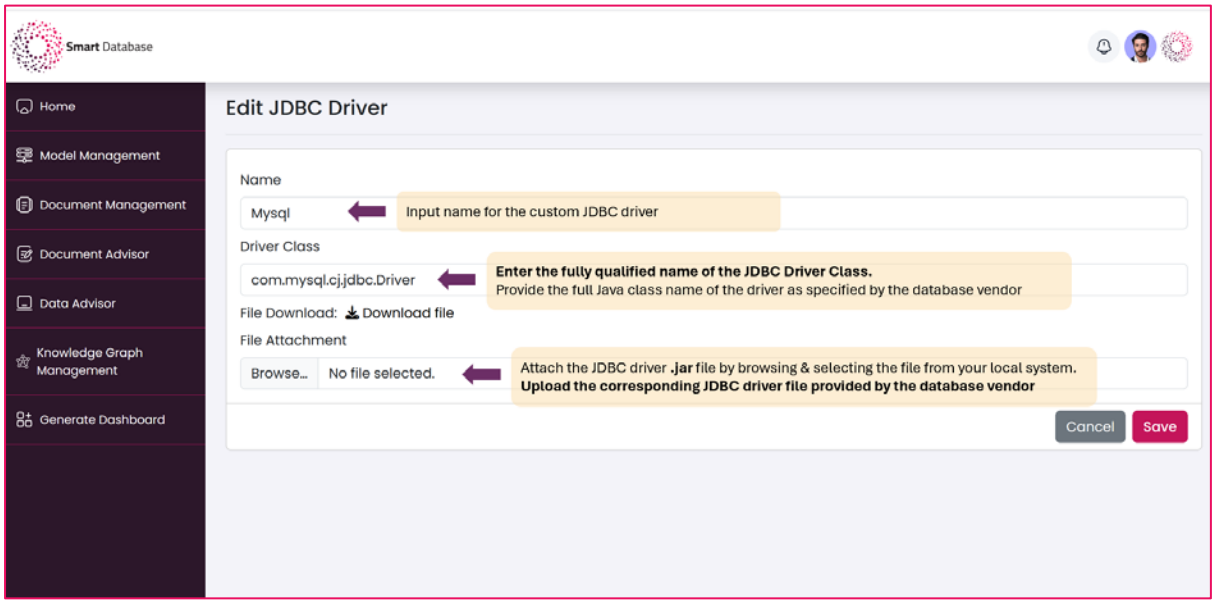


Figure 8: illustrates the Add JDBC Driver interface and highlights the required fields for registering a custom JDBC driver in SmartDB.

The Add JDBC Driver form contains the following fields:

1. Name

This field defines the **driver name** as it will appear within SmartDB.

- The name should clearly represent the database technology (e.g., *Oracle Custom*, *MongoDB Enterprise*, *Legacy ERP DB*).
- This name will later appear in the Select JDBC Driver dropdown when creating database connections.

2. Driver Class

This field specifies the fully qualified JDBC driver class name.

- The driver class is provided by the database vendor.
- It is required for SmartDB to load and initialize the JDBC driver correctly.

Example:

- com.mysql.cj.jdbc.Driver
- org.postgresql.Driver

3. File Attachment

This field allows the administrator to upload the **JDBC driver file** (typically a .jar file).

- The uploaded file contains the actual implementation of the JDBC driver.
- Only valid and compatible JDBC driver files should be uploaded.
- The driver file must match the specified Driver Class.

Registers the new JDBC driver in the SmartDB platform. Once saved, the driver becomes available for selection when configuring new JDBC database connections.

2.1.2.2.3 Administrative Notes

- Only users with administrative privileges can add or delete JDBC drivers.
- Adding custom JDBC drivers enables SmartDB to support a wider range of SQL and NoSQL databases beyond the predefined drivers.
- Removing a JDBC driver should be performed with caution, as existing database connections may depend on it.

2.1.3 Test Connection

The Test Connection feature allows users to validate the database connectivity for any configured JDBC connection within the SmartDB platform.

Users can test a database connection by selecting the Test Connection option from the Actions icons bar. This operation verifies that the database is reachable, the connection parameters are correct, and SmartDB services can successfully communicate with the database before it is used for analytics, advisory, or machine learning operations.

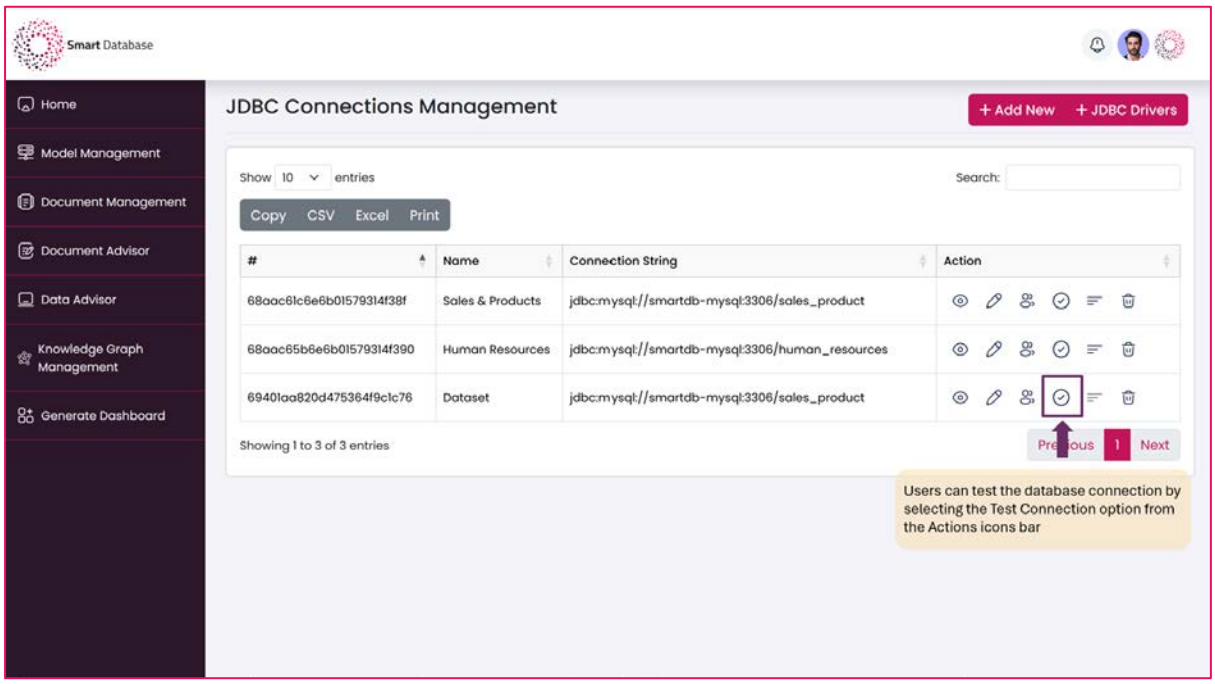


Figure 9: Test Connection process showing database connectivity validation in SmartDB

As a result of clicking the Test Connection option from the Actions icons bar, the system displays the database connectivity status, indicating whether the connection has been successfully established or not. The following figures illustrate both scenarios: a failed database connection status and a successful database connection status.

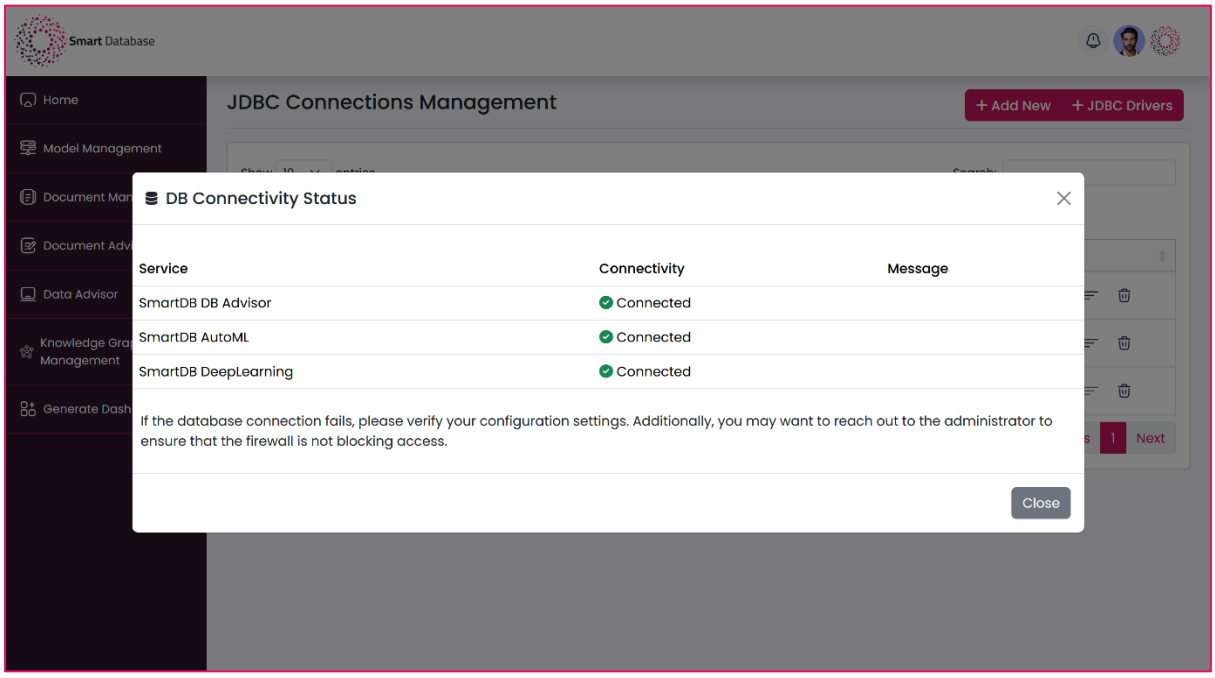


Figure 10: Test Connection process showing successful database connectivity status

As shown in **Figure 10**, this dialog provides a real-time status of the database connection across key SmartDB services, including:

- **SmartDB DB Advisor** – Validates connectivity for conversational and natural language database interactions
- **SmartDB AutoML** – Confirms availability of the database for machine learning and predictive analytics workflows
- **SmartDB Deep Learning** – Ensures the database can be accessed for advanced AI and deep learning operations

Each service displays a **Connected** status when the database connection is successfully established. A successful result confirms that the database configuration, credentials, and network access are correctly set and that SmartDB services can communicate with the database without issues.

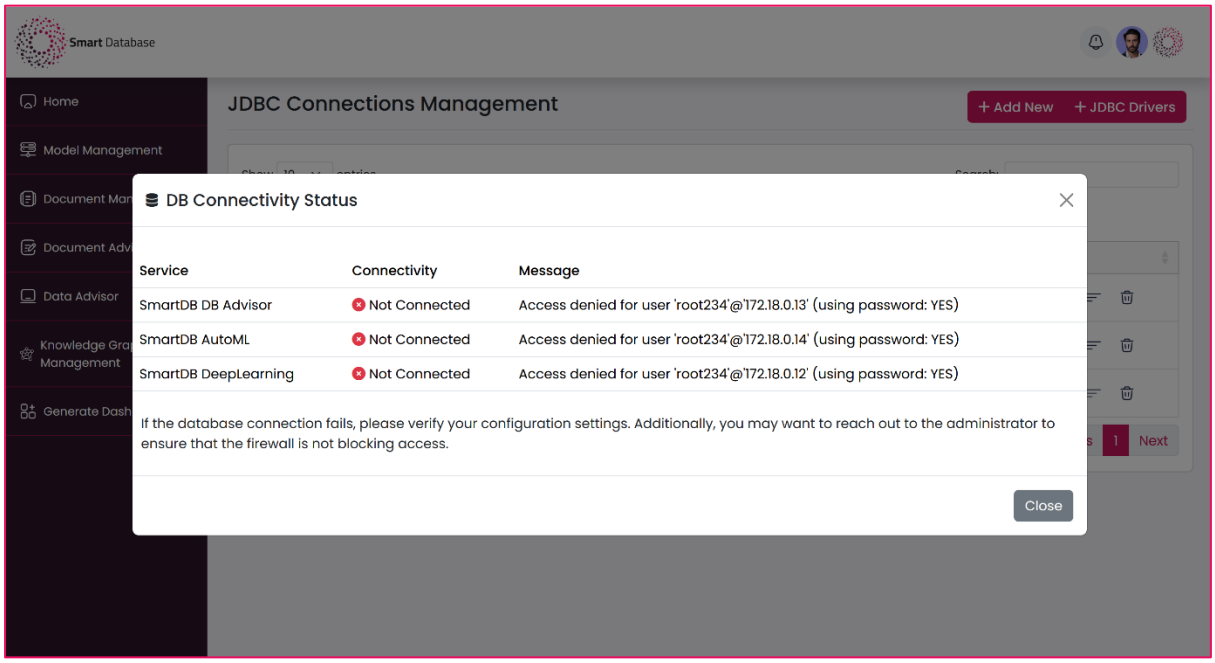


Figure 11: Test Connection process showing database connectivity failure status

If the database connection fails as shown in **figure 11**, users are advised to review the connection parameters, verify authentication details, and ensure that network and firewall rules allow access between SmartDB and the database server.

2.1.4 User Management (Assign Access to JDBC Connection)

The User Management section allows administrators to control which users can access specific JDBC connections and use SmartDB capabilities such as Data Advisory, Machine Learning (AutoML), and Knowledge Graph features.

As shown in the figure above, this page enables administrators to assign users to a selected JDBC connection, ensuring that access to enterprise data is governed, secure, and aligned with organizational roles and permissions.

Purpose of User Management

User Management ensures that:

- Only authorized users can view and query connected databases
- Access to sensitive data is restricted based on user roles
- SmartDB features such as natural language data advisory, ML model creation, and knowledge graph exploration are available only to permitted users

As shown in Figure 12, administrators can manage user access to a JDBC connection by clicking the Users icon in the Actions bar within the JDBC Connections Management page..

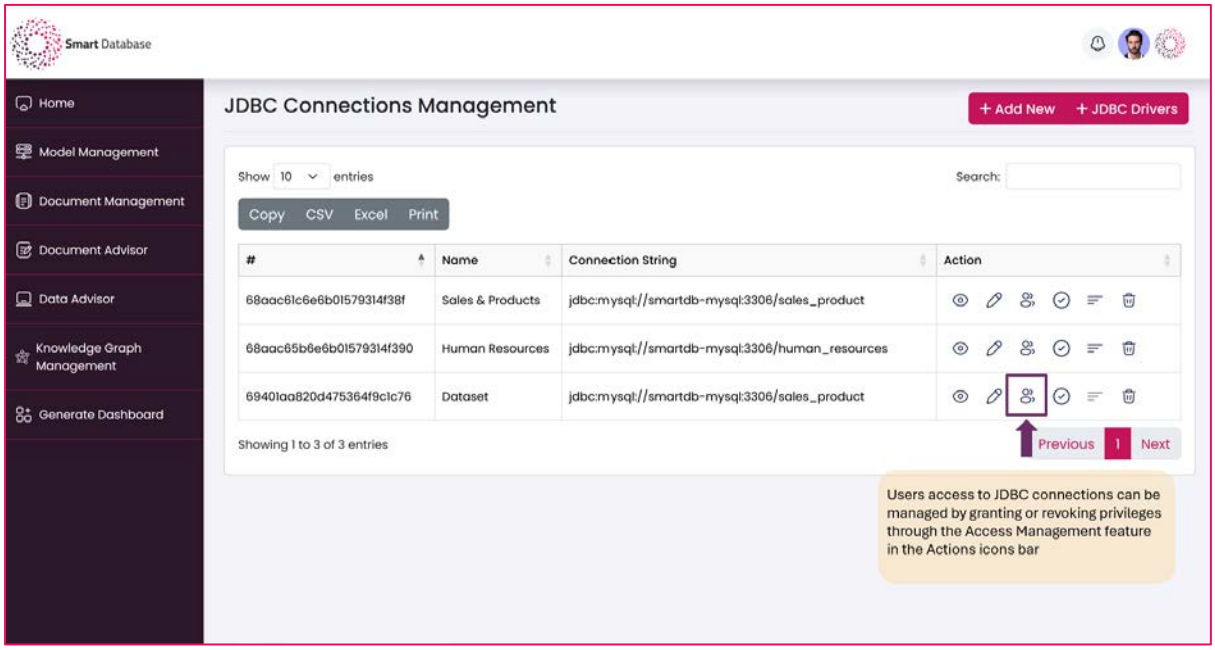


Figure 12: User Management Access for Managing JDBC Connection Permissions

2.1.4.1.1 Adding a User to a JDBC Connection

To grant a user access to a specific JDBC connection:

1. Navigate to **User Management** from the JDBC Connections context.
2. Use the **Select User** dropdown to choose an existing SmartDB user.
3. Click **Add** to assign the selected user to the JDBC connection.

Once added, the user will be able to access the associated database and utilize the enabled SmartDB capabilities based on their assigned permissions. Figure 13 demonstrates the user permissions interface for assigning and managing user access

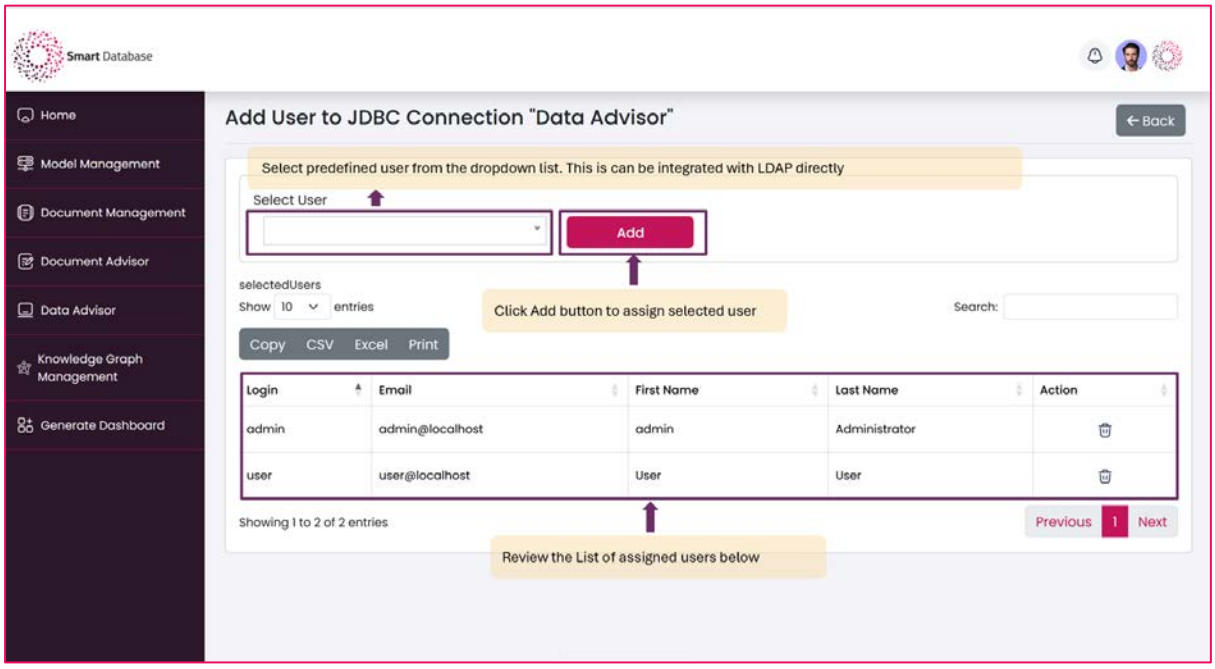


Figure 13: User Permission Screen for Adding and Managing User Access

2.1.4.1.2 Administrative Notes (Impact on SmartDB Capabilities)

Users assigned to a JDBC connection can:

- View and explore database schemas and datasets
- Interact with the database using **natural language queries** through Data Advisory
- Build and train **machine learning models** using connected data
- Generate and explore **Knowledge Graphs** derived from database content

Users who are not assigned will not be able to access or interact with the database through SmartDB (Smart Database).

2.2 Identity & Access Management Configuration (LDAP / Directory Services)

SmartDB (Smart Database) supports integration with enterprise directory services using LDAP (Lightweight Directory Access Protocol) to enable centralized authentication and user management. LDAP configuration allows organizations to manage SmartDB user access through their existing directory (such as Active Directory or any LDAP-compliant service), reducing manual user administration and enforcing consistent security policies.

When LDAP is enabled, SmartDB can authenticate users against the directory, synchronize users and groups (based on configured search rules), and map directory groups to SmartDB roles and privileges. This ensures that only authorized users can access SmartDB modules such as Data Advisory, AutoML, and Knowledge Graph, etc, according to the organization's access governance model.

Purpose of LDAP Configuration

LDAP configuration in SmartDB (Smart Database) is used to:

- Centralize user authentication using corporate directory credentials
- Reduce manual user creation and maintenance within SmartDB
- Synchronize users and groups from the directory (optional, based on policy)
- Enable role and privilege assignment through group-based mapping
- Support audit and compliance requirements through consistent identity control

2.2.1 Accessing LDAP Configuration

Administrators can access the LDAP configuration settings through the SmartDB Administration menu. From the main dashboard, navigate to the top-right menu and select User Management, then proceed to the LDAP Configuration option to manage directory integration settings.

As shown in **Figure 14**, the User Management menu provides access to LDAP configuration, allowing administrators to configure, enable, and manage directory-based authentication for SmartDB users.

This section is restricted to users with administrative privileges and is used to integrate SmartDB with enterprise directory services such as Active Directory or other LDAP-compliant systems.

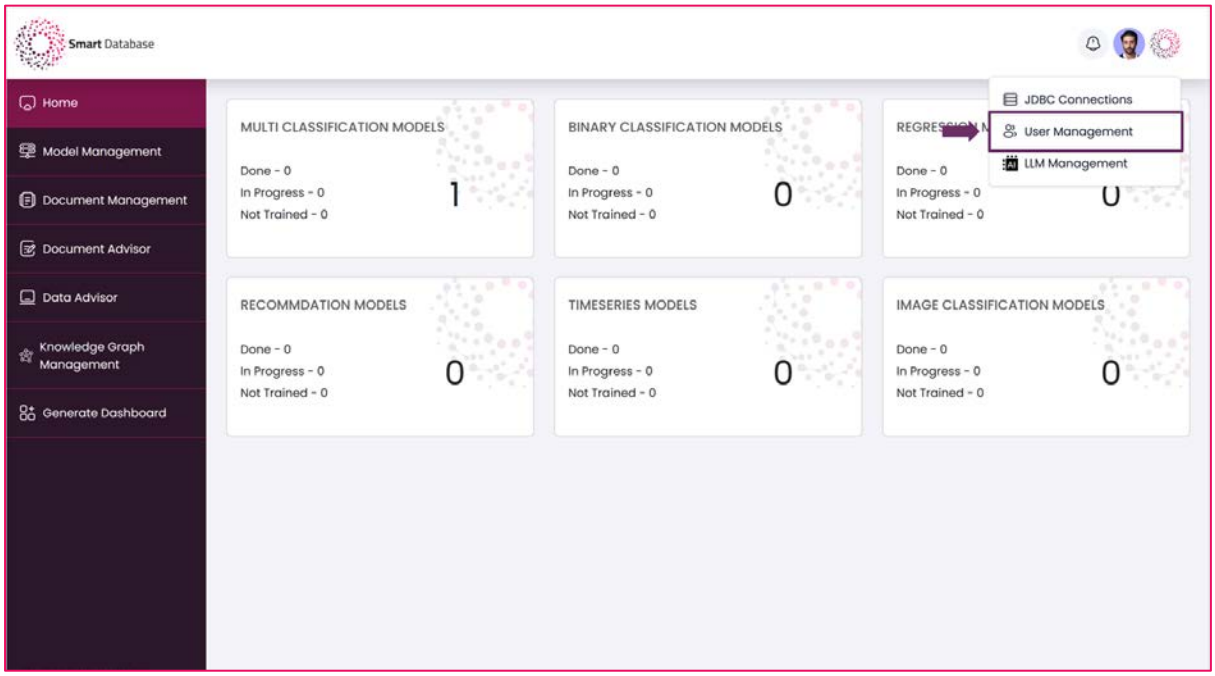


Figure 14: Accessing LDAP configuration through the User Management menu

2.2.2 User Management – LDAP Configuration Access

The User Management page serves as the central control point for managing user accounts, authentication sources, and access roles within the SmartDB platform. From this page, administrators can manage both manually created users and directory-based users synchronized from LDAP.

As shown in **Figure 15**, the top-right action bar provides two key options:

- **Add New:** Allows administrators to create local (manual) user accounts directly within SmartDB. These users can be assigned roles and permissions without relying on an external identity provider.
- **+ LDAP Config:** Enables administrators to configure LDAP integration. This option is used to connect SmartDB to an enterprise directory service (such as Active Directory or OpenLDAP) and automatically load users and organizational structure into the platform.

Once LDAP synchronization is completed, all users retrieved from the LDAP directory are listed in the User Management table, alongside locally created users. For each user, the table displays key information such as login name, email address, source name, assigned roles (authorities), and activation status. Administrators can activate or deactivate users and manage their access to SmartDB features, including Data Advisory, Machine Learning, and Knowledge Graph modules.

This unified view ensures centralized governance over user identities and permissions while supporting both enterprise-grade LDAP authentication and flexible local user management within SmartDB.

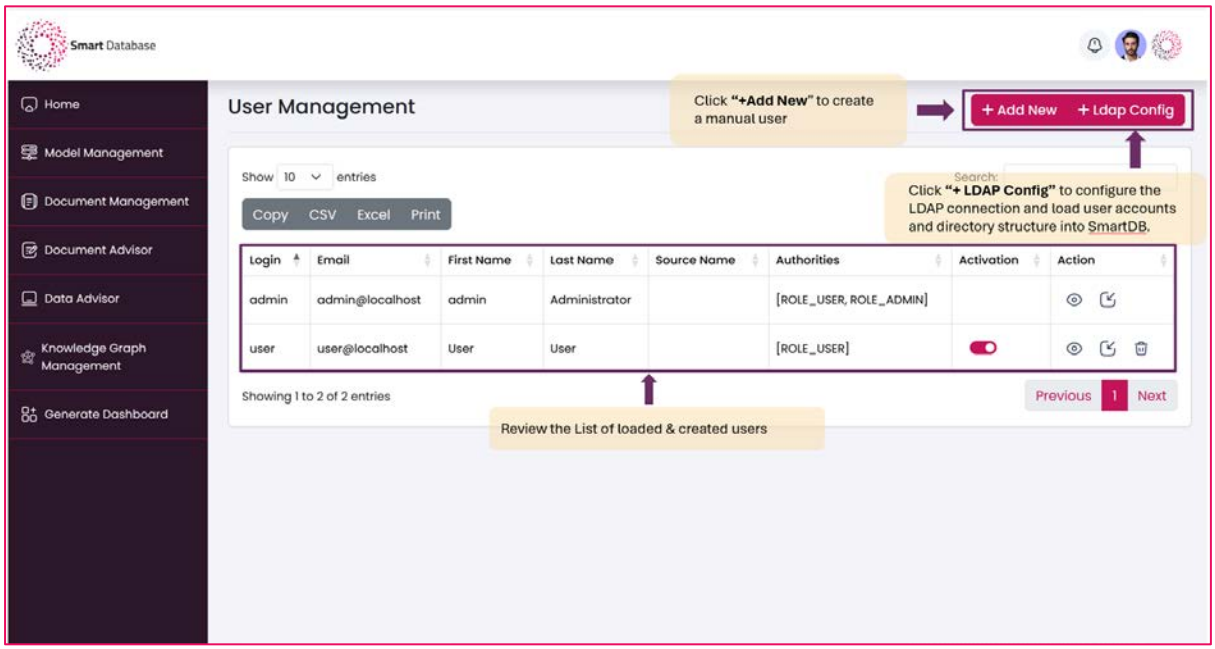


Figure 15: User Management page showing access to manual user creation and LDAP configuration

This approach allows SmartDB (Smart Database) to support hybrid user management, combining centralized enterprise authentication via LDAP with flexible manual user creation when required.

2.2.3 LDAP Management Page

The LDAP Management page allows administrators to configure, manage, and monitor LDAP directory integrations within the Smart Database platform. This page serves as the central location for defining LDAP connections that enable centralized authentication, user synchronization, and organizational structure loading from enterprise directory services.

From this page, administrators can view all configured LDAP connections in a structured table that displays key configuration details, including Host, Protocol, Port, and Base DN. Once configured, these LDAP connections are used to automatically load users, groups, and directory hierarchies into SmartDB, enabling seamless access control and identity management.

To create a new LDAP configuration, administrators can click the "+ Add New" button located at the top of the page. This action opens the LDAP configuration form, where connection parameters can be defined and validated before loading users and directory structures into the system.

As shown in **Figure 16**, the **"+ Add New"** button clearly indicates where administrators can initiate the creation of a new LDAP configuration and load the LDAP directory into SmartDB.

Once configured, all users and directory structures loaded from LDAP will automatically appear in the User Management section and can be assigned appropriate roles and permissions within SmartDB (Smart Database).

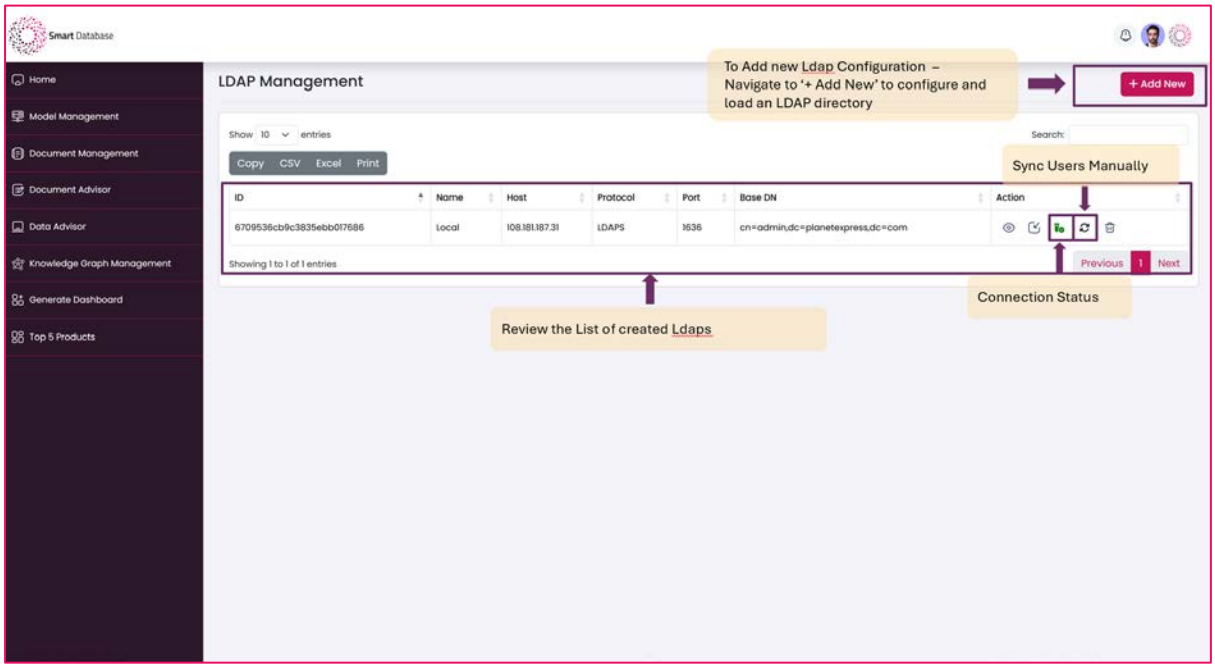


Figure 16: LDAP Management Interface for Creating LDAP Connection

The LDAP Management page ensures centralized governance over directory integrations, simplifies enterprise user onboarding, and supports secure, scalable access to SmartDB capabilities across all modules.

2.2.4 LDAP Configuration – Add LDAP Config Page

The Add LDAP Config page allows administrators to configure and integrate an external LDAP or Active Directory service with the Smart Database platform. This configuration enables centralized user authentication, automatic user provisioning, and synchronization of directory structures and user attributes into SmartDB (Smart Database).

As shown in **Figure 17**, this page provides a comprehensive set of configuration fields that define how SmartDB (Smart Database) connects to the LDAP server, retrieves users, maps user attributes, and assigns roles.

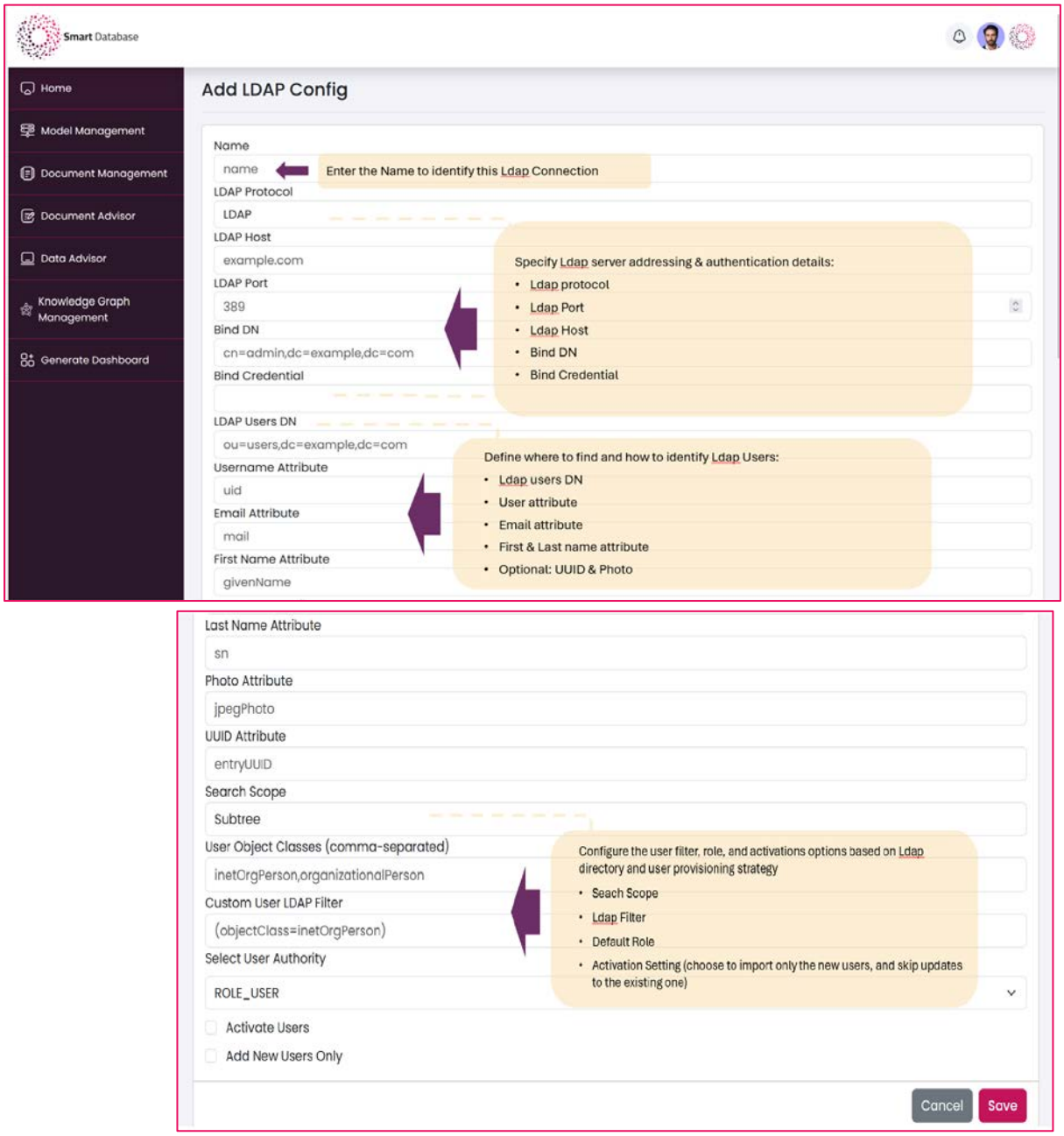


Figure 17: Add LDAP Configuration Page Showing LDAP Connection and User Mapping Settings

2.2.4.1 LDAP Configuration Fields

1. **Name**

   A friendly name for the LDAP configuration. This is used to identify the LDAP connection within SmartDB.

2. **LDAP Protocol**

   Specifies the protocol used to connect to the directory service, such as **LDAP** or **LDAPS** (secure LDAP).

3. **LDAP Host**

   The hostname or IP address of the LDAP server (e.g., ldap.example.com).

4. **LDAP Port**

   The port used for LDAP communication. Common values include:

   - 389 for LDAP
   - 636 for LDAPS

5. **Bind DN**

   The distinguished name of the service account used by SmartDB to bind and query the LDAP directory (e.g., cn=admin,dc=example,dc=com).

6. **Bind Credential**

   The password for the Bind DN account. This credential is securely stored and used only for directory access.

7. **LDAP Users DN**

   Defines the base distinguished name where user entries are located in the directory (e.g., ou=users,dc=example,dc=com).

8. **Username Attribute**

   Specifies the LDAP attribute used as the login username in SmartDB (e.g., uid).

9. **Email Attribute**

   The LDAP attribute mapped to the user's email address (e.g., mail).

10. **First Name Attribute**

    The LDAP attribute representing the user's first name (e.g., givenName).

11. **Last Name Attribute**

    The LDAP attribute representing the user's last name (e.g., sn).

12. **Photo Attribute**

    Optional attribute used to retrieve the user's profile image (e.g., jpegPhoto).

13. **UUID Attribute**

    A unique identifier attribute used to uniquely track users across synchronizations (e.g., entryUUID).

14. **Search Scope**

    Defines how deep SmartDB searches within the directory structure:

    - **Subtree** – Searches all child entries under the base DN
    - **One Level** – Searches only direct children

15. **User Object Classes**

    A comma-separated list of LDAP object classes that represent user accounts (e.g., inetOrgPerson, organizationalPerson).

16. **Custom User LDAP Filter**

    An optional LDAP filter used to restrict which users are imported (e.g., (objectClass=inetOrgPerson)).

17. **Select User Authority**

    Defines the default role assigned to users imported from LDAP (e.g., ROLE_USER).

18. **Activate Users**

    When enabled, users imported from LDAP are automatically activated in SmartDB.

19. **Add New Users Only**

    When selected, only new users from LDAP are added, and existing users are not updated.

2.2.4.2 Saving the Configuration

After completing all required fields, the administrator can click **Save** to store the LDAP configuration. Once saved, SmartDB connects to the LDAP directory and loads users and directory structures according to the defined settings. Imported users will then appear in the **User Management** section and can be granted access to SmartDB modules such as Data Advisory, Machine Learning, and Knowledge Graph.

2.3 Large Language Model (LLM) Configuration

The Large Language Model (LLM) Configuration module in SmartDB (Smart Database) enables administrators to register, configure, and manage AI language models that power natural-language capabilities across the platform. These models are used by features such as Data Advisory, Document Advisor, SmartQuery, and other GenAI-driven services.

In addition to integrating local or external LLM services, SmartDB allows organizations to download and deploy any open-source Large Language Model and embed it directly into the platform. These models can be hosted within the organization's own infrastructure and connected to SmartDB through supported local runtime engines or APIs. This approach enables full control over the model lifecycle, including versioning, fine-tuning, performance optimization, and security enforcement, while ensuring complete data sovereignty and compliance with internal governance policies.

SmartDB is designed with a **model-agnostic architecture**, allowing organizations to flexibly deploy and connect LLMs based on their security, compliance, performance, and sovereignty requirements.

2.3.1 Supported LLM Deployment Models

SmartDB (Smart Database) supports both local (on-premise) and external (cloud-based) LLM integrations:

a) Local / On-Premise LLM Integration

Organizations can deploy LLMs within their own infrastructure and connect them directly to SmartDB. This approach is suitable for environments requiring full data sovereignty and offline operation.

Supported local deployment options include:

- LLMs hosted using **Ollama** or similar local model servers
- Self-hosted open-source models (e.g., LLaMA, Mistral, Falcon, Mixtral)
- GPU or CPU-based inference environments

Local LLMs communicate with SmartDB through secure internal APIs, ensuring that sensitive data never leaves the organization's network.

Note: Installation guidelines are covered in the Installation Guide document.

b) External / Cloud-Based LLM Integration

SmartDB also supports integration with external LLM providers to leverage advanced reasoning and continuously updated models.

Examples include:

- **ChatGPT / OpenAI APIs**
- **GitHub-hosted or marketplace LLMs**
- Other third-party LLM services accessible via REST or API endpoints

This option enables rapid deployment and access to state-of-the-art models while maintaining controlled data access and usage policies.

2.3.2 Accessing LLM Configuration

The LLM Configuration section enables administrators to manage, integrate, and control Large Language Models used by the Smart Database platform. Through this interface, organizations can connect both local and external LLM engines, ensuring flexibility, security, and alignment with enterprise or government deployment requirements.

Administrators can access the LLM Configuration module from the Model Management menu by selecting LLM Management, as shown in **Figure 18**.

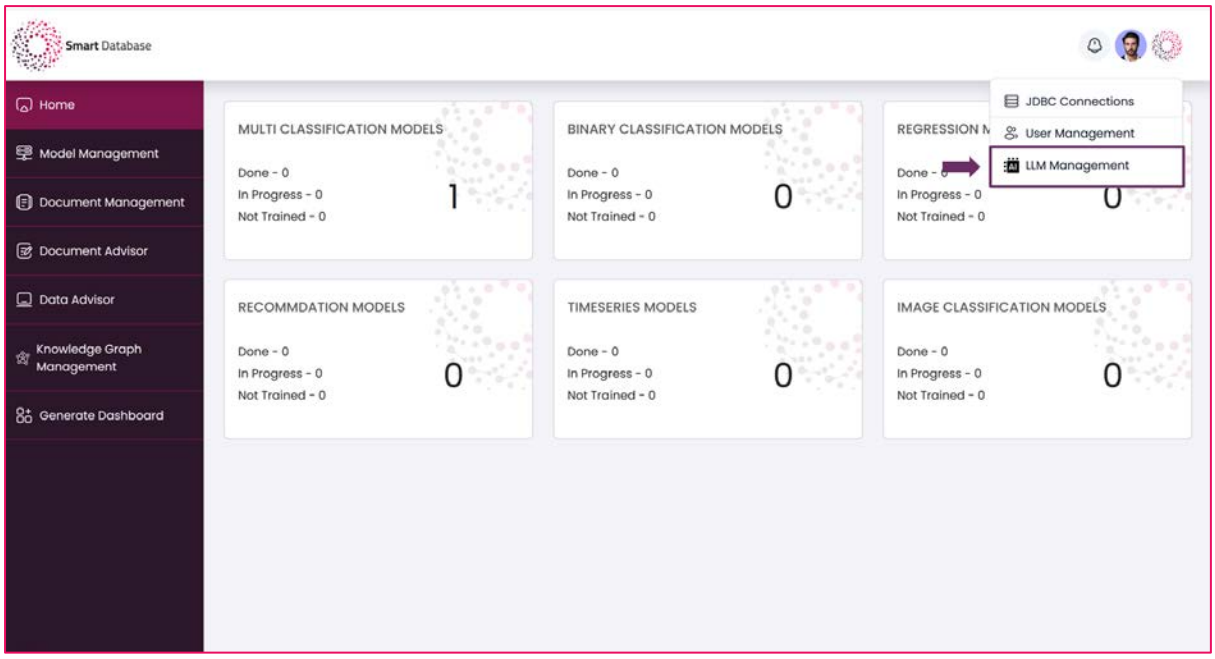


Figure 18: Accessing the Large Language Model (LLM) Configuration Module from Model Management

Supported LLM Integration Options

Smart Database supports multiple LLM deployment approaches:

- **Local LLM Deployment**
  Organizations can deploy and run LLMs within their own environment using local inference engines such as Ollama or similar frameworks. This approach is ideal for sovereign, air-gapped, or high-security environments.
- **Open-Source LLM Integration**
  Any supported open-source LLM can be downloaded and embedded directly into the Smart Database module, enabling full control over model behavior, data privacy, and performance tuning.
- **External LLM Services**
  Smart Database can integrate with external LLM providers such as ChatGPT, GitHub-hosted models, and other third-party AI services through secure APIs. This allows organizations to leverage advanced commercial models without local deployment overhead.

Note: Detailed installation steps, prerequisites, and environment preparation are provided in the **Installation Guide** document.

2.3.3 LLM Management

The LLM Management page provides administrators with a centralized interface to manage all configured Large Language Models (LLMs) within the Smart Database platform. This page displays a consolidated list of both local and external LLM integrations currently available for use across SmartDB features such as Data Advisory, Document Advisor, Knowledge Graph, and AI-driven analytics.

As shown in **Figure 19**, administrators can review all configured LLMs, including their unique identifiers, integration type, and assigned names. The platform supports multiple LLM sources, such as locally deployed models (e.g., via Ollama) and externally integrated models (e.g., GitHub-hosted or third-party APIs).

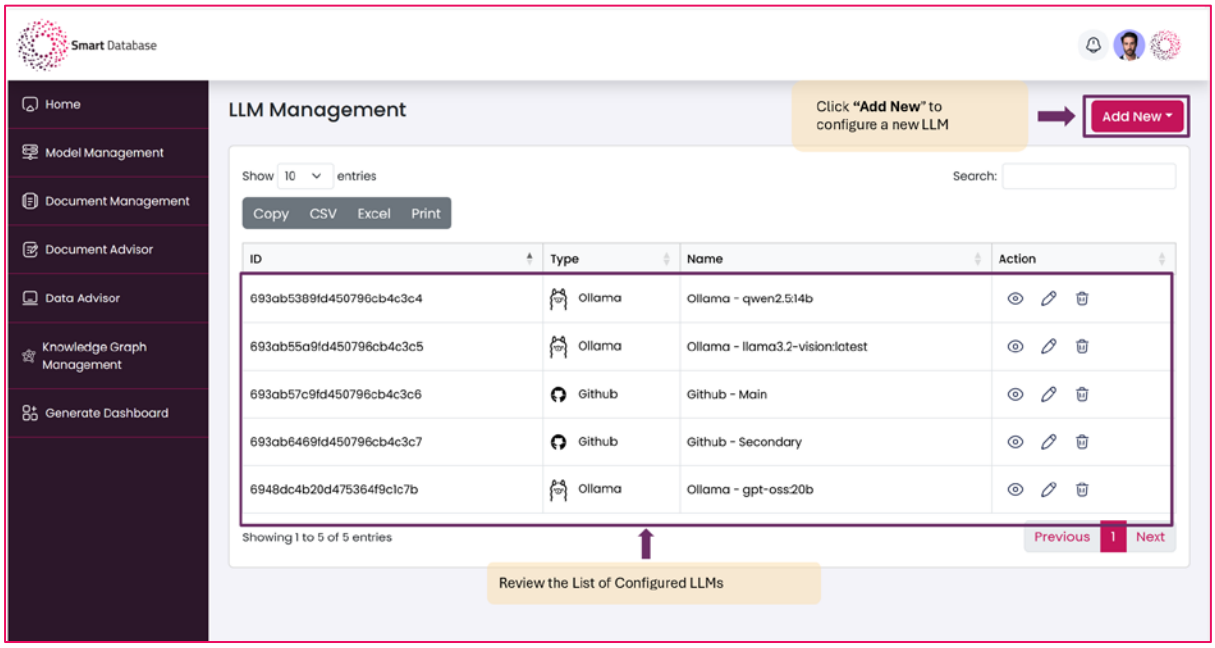


Figure 15: LLM Management Page Showing Configured Local and External Large Language Models

2.3.3.1 Key Capabilities

- **View Configured LLMs**

  The table lists all registered LLMs with details such as:

  - LLM ID
  - Integration type (e.g., Ollama, GitHub, external providers)
  - Model name and version

- **Add New LLM Configuration**

  Administrators can click the **"Add New"** button to configure and register a new LLM, whether deployed locally or accessed through an external service.

- **Manage Existing LLMs**

  From the **Actions** column, administrators can:

  - View LLM configuration details
  - Edit existing LLM settings
  - Remove an LLM from the platform

This centralized management ensures full governance, flexibility, and control over which language models are available to Smart Database users and services.

2.3.4 Adding a New Large Language Model (LLM)

The LLM Management page allows administrators to add and configure new Large Language Models that can be used across the Smart Database platform. From this page, SmartDB supports both on-premise (local) LLM deployments and external (cloud-based) LLM integrations, enabling full flexibility based on security, compliance, and performance requirements.

As shown in **Figure 19**, administrators can initiate the LLM configuration process by clicking the "**Add New**" button. Upon clicking this option, a dropdown menu appears allowing the administrator to select the type of LLM to configure.

The platform supports multiple LLM providers and deployment models, including but not limited to:

- **Ollama** (local / on-premise LLM deployment)
- **GitHub** (custom or hosted LLM services)
- **OpenAI**
- **Hugging Face**
- **Microsoft Azure OpenAI**
- **Amazon Bedrock**
- **Google Gemini**
- **Anthropic**

This selection step defines whether the LLM will run within the customer's local environment or be accessed through an external service provider. Once the model type is selected, the system guides the administrator to the corresponding configuration screen to complete connection details, credentials, and model-specific parameters.

Note: Detailed installation and deployment instructions for on-premise LLMs not in the pre-defined list (such as any open-source models) are provided separately in the **Installation Guide** document.

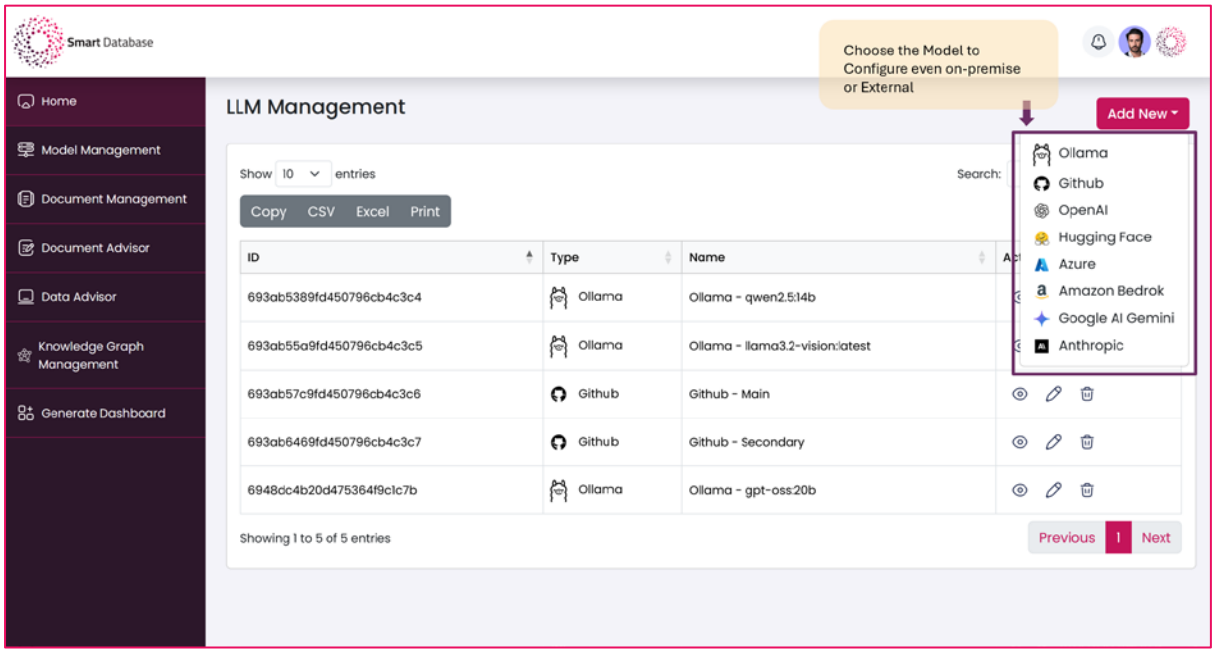


Figure 19: Selecting the LLM Type for On-Premise or External Model Configuration

2.3.5 Configuring an LLM Connection

This page enables administrators to configure and manage Large Language Model (LLM) connections for both on-premise and external deployments within the Smart Database platform.

Using this interface, users can add an on-premise LLM connection by linking to a locally hosted runtime such as Ollama, allowing the platform to consume models running inside the organization's infrastructure. This option is suitable for environments requiring enhanced data privacy, offline operation, or sovereign AI deployment.

In addition, the platform supports external LLM integrations, such as ChatGPT (OpenAI), GitHub Models, and other cloud-based providers, enabling easy access to managed LLM services.

As illustrated in **Figure 20**, administrators can create a new LLM connection by providing the following configuration details:

- **Name**

  A logical name to identify the LLM connection within the Smart Database platform.
- **URL**

  The endpoint URL of the locally running (for example, http://0.0.0.0:11434). This endpoint is used by SmartDB to communicate with the on-premise LLM runtime.

- **Model**

  The specific LLM model identifier hosted (for example, llama3.1:70b). The selected model will be used for inference and reasoning tasks across SmartDB features such as Data Advisory, Document Advisor, and Knowledge Graph processing.

- **Keep Alive**

  An optional setting that keeps the model loaded in memory to reduce startup latency and improve response time for frequent requests.

Once all required fields are completed, the administrator can click **Save** to register the LLM connection or **Cancel** to discard the configuration. After saving, the LLM becomes available for selection and use across supported Smart Database modules.

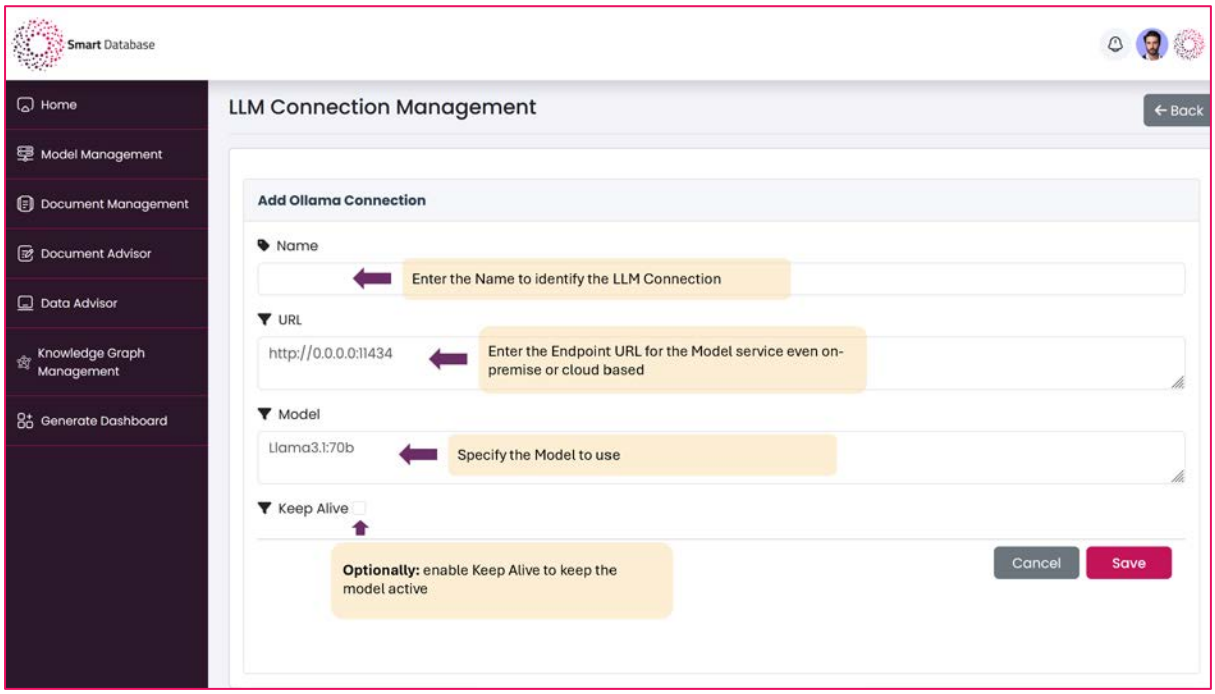
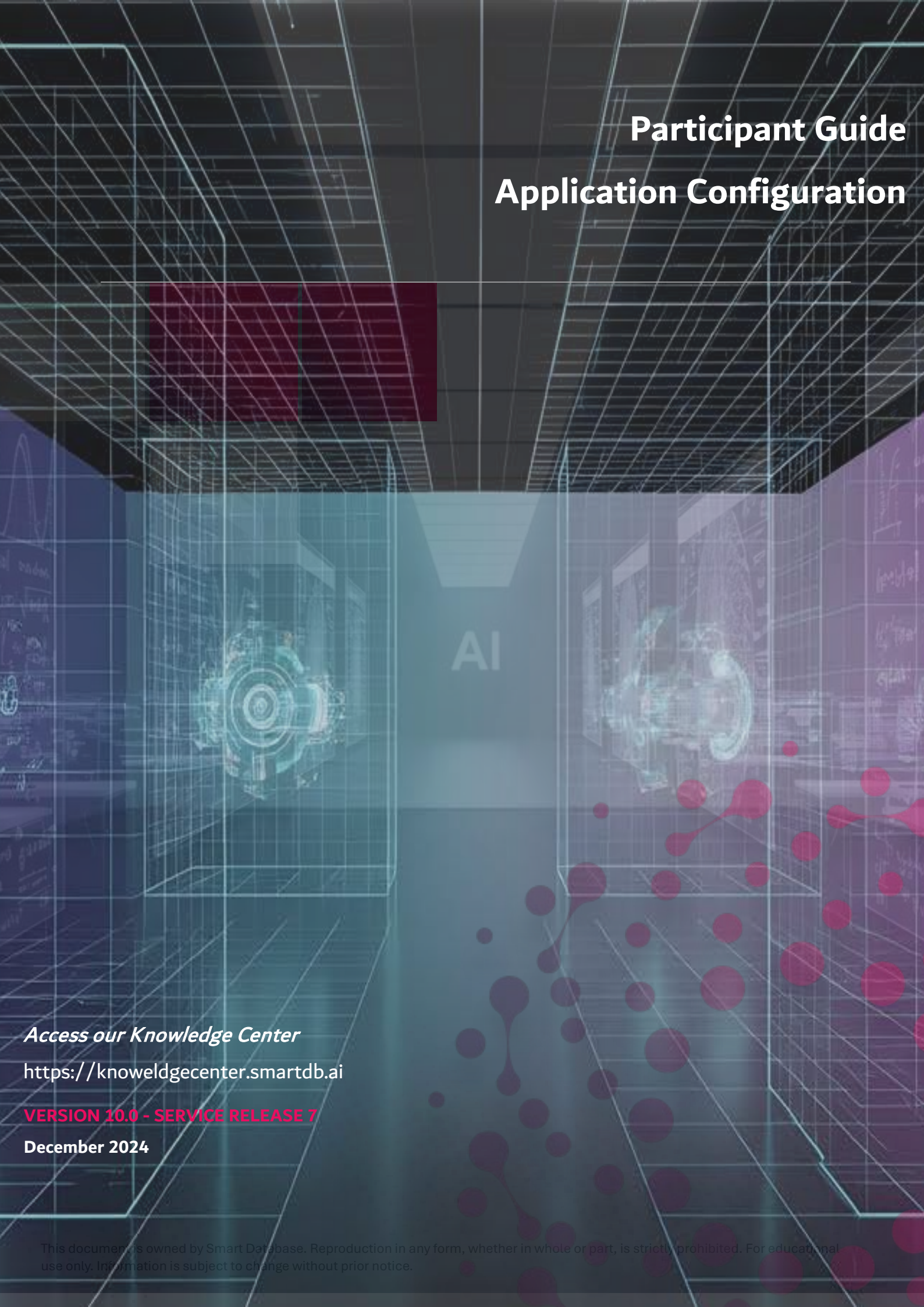


Figure 20: Configuring an On-Premise & external LLM Connection

Note: Installation, model download, and runtime setup instructions for Ollama and other open-source LLMs are documented separately in the **Installation Guide**.

Participant Guide
Application Configuration



Access our Knowledge Center

https://knoweldgecenter.smartdb.ai

VERSION 10.0 - SERVICE RELEASE 7

December 2024