Column Configuration Documentation

Overview

The Column Configuration section is a critical administrative feature that allows administrators to manage which columns in database tables are editable by makers. It enables fine-grained control over data modification access at the column level.

Purpose

- Specify editable columns per table for makers
- Protect sensitive columns by restricting edit access
- Provide an intuitive UI for managing column-level permissions
- Support renamed columns for improved clarity and readability

Key Features

1. Table Selection
    - Dropdown list of available tables
    - Excludes system or restricted tables
    - Searchable and well-organized interface
2. Column Permission Management
    - Toggle edit access for each column individually
    - Batch reset option to mark all columns as non-editable
    - Pagination for tables with many columns
    - Visual indicators for permission states
3. Column Display Names
    - Supports custom display names
    - Auto-formats raw column names for readability
    - Syncs with column rename configurations

Workflow

Initial Load

- Loads available tables upon section access
- System and restricted tables are excluded
- Dropdown populated with eligible table names

Table Selection Process

- On table selection:

    - Fetches all columns from selected table
    - Retrieves current column-level permissions
    - Loads any custom column display names
    - Columns are shown in a paginated grid

Permission Management

- Each column displays:
    - Original or renamed column name
    - Editable/non-editable status
    - Toggle switch for permission change
- Batch Controls:
    - Reset to mark all as non-editable
    - Save to persist changes

API Interactions

Table Data Retrieval

- Endpoint: /table
- Fetches list of available tables
- Requires admin token
- Returns table names and metadata

Column Data Retrieval

- Endpoint: /fetchcolumn
- Retrieves columns for the selected table
- Requires admin token
- Returns column names and data types

Permission Management

- Endpoint: /ColumnPermission
    - GET: Fetch current column permissions
    - PUT: Save updated permissions
- Requires admin token

Column Renaming

- Endpoint: /renamed/{tableName}

- Retrieves or sets custom column display names
- Requires admin token
- Returns mapping of original to renamed columns

Security Considerations

Authentication

- All requests require a valid admin token
- Session validation enforced
- Token required in request headers

Permission Persistence

- Changes logged and saved immediately
- Makers' access updated in real-time
- Secure backend storage for permission configs

User Guide

Accessing the Configuration

- Log into the admin dashboard
- Select "Column Configuration"
- Ensure admin role is assigned

Managing Column Permissions

- Choose a table from the dropdown
- Review column list
- Use toggle to grant or restrict edit access
    - ON: Editable by makers
    - OFF: Read-only
- Use pagination for navigation
- Click "Reset" to revoke all edit access
- Click "Save" to apply changes

Best Practices

- Regularly audit column permissions
- Restrict sensitive data fields
- Rename columns for better understanding
- Keep a log of permission changes

Impact on Makers

- Editable columns shown with input fields
- Read-only columns displayed without edit controls
- Renamed columns appear in the maker interface
- Changes are applied instantly after save

Troubleshooting

Permission Changes Not Saving

- Check if token is expired or invalid
- Ensure a stable network connection
- Validate API response status codes

Tables Not Loading

- Confirm database is online
- Check admin access rights
- Test table-fetch API

Column Display Issues

- Review renamed column configurations
- Test UI on different browsers
- Confirm display name mappings exist

Maintenance and Updates

Regular Tasks

- Review column permissions per table
- Update renamed column values as needed
- Configure permissions for newly added tables