AppSmith

Appsmith ist die All-in-One-Lösung für deinen Bedarf an internen Tools. Es ist eine quelloffene Low-Code-Plattform, mit der du deine Softwareanwendungen an einem Ort erstellen, hosten und verwalten kannst.

Mit Appsmith kannst du benutzerdefinierte Anwendungen erstellen, mit beliebigen Datenquellen verbinden und in wenigen Minuten eine schöne Benutzeroberfläche erstellen. Das spart Zeit, denn du brauchst keine Entwickler/innen oder IT-Mitarbeiter/innen mehr, die diese Anwendungen normalerweise getrennt voneinander erstellen müssten. Außerdem sorgt es für mehr Sicherheit, indem es sicherstellt, dass nur befugte Nutzer/innen Zugriff auf bestimmte Teile der Funktionen einer App haben - also keine versehentlichen Lecks in den Systemen der Konkurrenz mehr!

Außerdem kann Appsmith lokal oder auf deiner privaten Instanz mit Docker bereitgestellt werden.

Installation

Installation unter Debian/Ubunti via Ansible

Ansible auf dem Konfig Laptop/PC installieren

Siehe Ansible Buch Ansible Buch

Playbook vorbereiten.

Auf dem konfigurations PC im Home Verzeichnis ein neues Verzeichnis erstellen. In Unserem Beispiel installappsmith
Auf dem Server wo appsmith drauf soll muss sich mit schlüsseldateien angemeldet werden können.

Nun das Verzeichnis anlegen

```
cd ~
mkdir installappsmith
```

Falls nicht vorhanden git installieren

```
apt install git
```

Nun in dem Verzeichnis das Playbook downloaden

```
git clone https://github.com/appsmithorg/appsmith.git
cd ./appsmith/deploy/ansible/appsmith_playbook
```

In dem Verzeichnis gibt es eine Extra Vars Datei (appsmith-vars.yml)
in der kann das Isntallationsverezcihnis geändert werden. Wir lassen es bei Standard.
(~/appsmith) So mit wird im Home root Verzeichnis im Verzeichnis appsmith installiert.

Nun die Inventory Datei anlegen.

```
nano hosts.ini
```

Inhalt: ip user und Port natürlich auf eure Bedürfnisse anpassen.

```
[appsmith]
192.168.178.75  ansible_port=22 ansible_user=root
```

Playbook ausführen und zurücklehnen

```
ansible-playbook -i hosts.ini appsmith-playbook.yml --extra-vars "@appsmith-vars.yml"
```

Ausgabe

```
PLAY [Configure the self-hosted server] ***************************************

TASK [Gathering Facts] ********************************************************
ok: [192.168.178.75]

TASK [setup-appsmith : Gather the package facts] ******************************
ok: [192.168.178.75]

TASK [setup-appsmith : include_tasks] *****************************************
included: /home/duffy/appsmithinstall/appsmith/deploy/ansible/appsmith_playbook/roles/setup-
appsmith/tasks/preflight.yml for 192.168.178.75
.....
RUNNING HANDLER [setup-appsmith : Successful installation message] *************
ok: [192.168.178.75] => {
    "msg": "Successful installation. Please open http://192.168.178.75/ to view your instance"
}

PLAY RECAP ********************************************************************
192.168.178.75             : ok=24   changed=11   unreachable=0    failed=0    skipped=4    rescued=0
ignored=1
```

Nun im Browser http://192.168.178.75/ öffnen

Entwicklung

Entwicklung

Einfache Tabelle mit Beziehungen

Beschreibung:

Wir haben 3 Tabellen maridb Tabellen

kunden

ID	vorname	nachname	firma

software

ID	name

lizenzen

ID	lizenznummer	kunde	software
		Verknüpft mit ID von Kunden	Verknüpfung mit ID von Software

Ziel ist es. in der Dritten Tabelle in der Spalte Kunde den Kundenname stehen zu Haben und nicht die ID
Der Kundenname soll bestehen aus Firma, Vorname, Nachname
Die Spalte Software soll den Softwarenamen enthalten und nicht die ID.

los gehts:

Tabellenwidgets vorbereiten

In Appsmith ein Tabellenwidget hinzufügen

Nun eine abfrage oder eine view erstellen.
Wir erstellen erstmal eine Abfrage

```
SELECT
    lizenzen.ID,
    lizenzen.lizenzschlüssel,
□□CONCAT(kunden.firma,'|',kunden.vorname,'|', kunden.nachname) AS kunde,
    software.name as software

FROM
    lizenzen
INNER JOIN
    kunden ON lizenzen.kunde = kunden.ID
INNER JOIN
□□software ON lizenzen.software = software.ID;
```

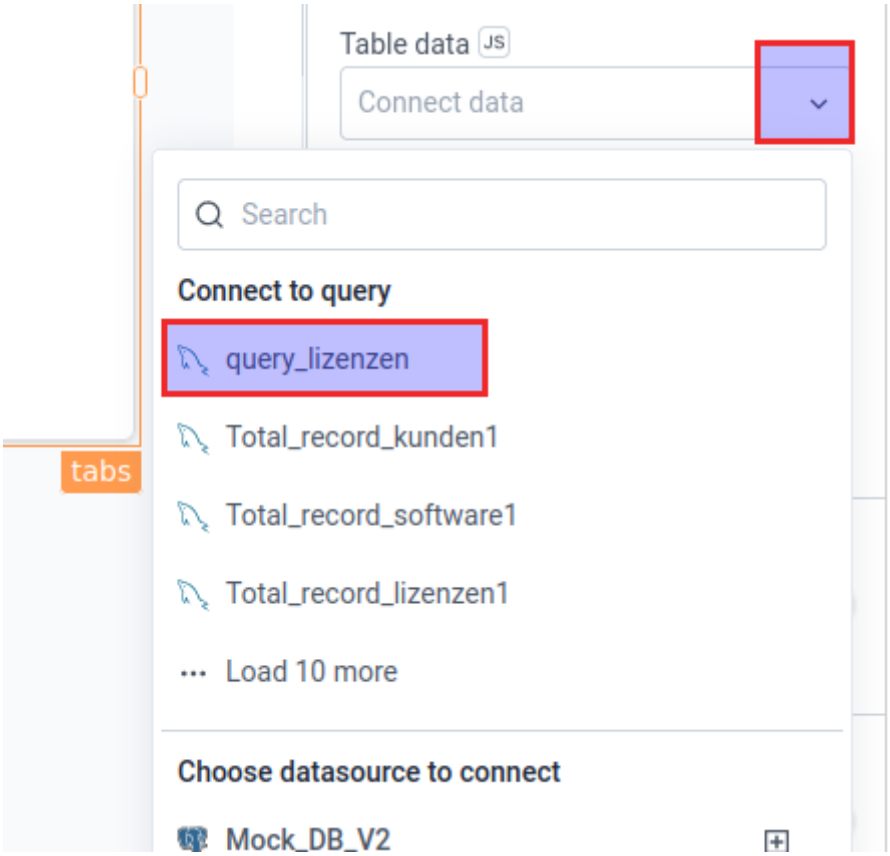
Nun mit Beschreibung

```
SELECT #folgende spalten auswählen, dies wird eine abfrage wie eine komplett neu gestallte tabelle view
erzeugt wird
    lizenzen.ID, #id von lizenzen
    lizenzen.lizenzschlüssel, #der lizenzschlüssel
□□CONCAT(kunden.firma,'|',kunden.vorname,'|', kunden.nachname) AS kunde, #hier wird ein komplett neuer strin
gebaut aus einer anderen tabelle uns als neue spalte eingefügt
    software.name AS software #hier wird der softwarename in eine neue Spalte namens software eingefügt
```

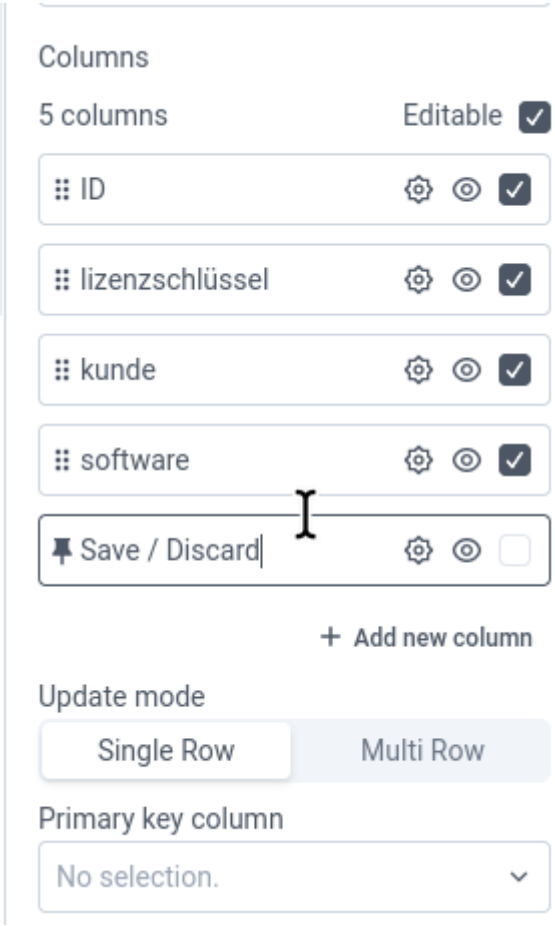
```
FROM #aus der tabelle lizenenzen das select
    lizenzen
INNER JOIN #spalte kunde mit der id aus der Tabelle kunden verknüpfen, die spalte kunde enthält den string. mit dem zusammen gestezten kundennamen. Mit der verknüpfung wird die id zugeordnet umd dann den dazugehörigen richtigen string aus der tabelle kunden zu holen damit  dann in der splalte kunde der richtige kundenname angezeigt wird
    kunden ON lizenzen.kunde = kunden.ID
INNER JOIN #spalte software mit der id aus der tabelle software verknüpfen, die spalte software enthält den string des namens der software. Mit der verknüpfung wird die id aus der software übergeben an die tabelle software um dann den dazugehörigen softwarenamen anzuzeigen
□□software ON lizenzen.software = software.ID;
```

Diese query nennen wir query_lizenzen

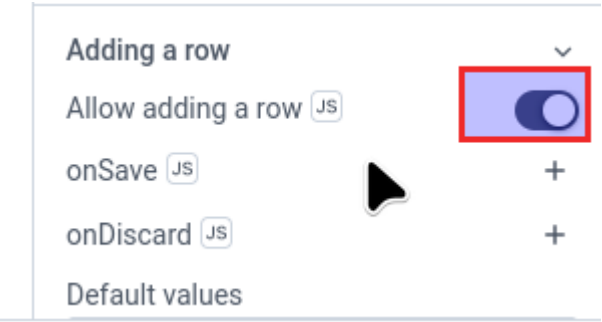
Nun gehen wir in die Eigenschaften des widget klciken auf den Pfeil bei connect Data und wählen unsere gerade erstellte query aus.



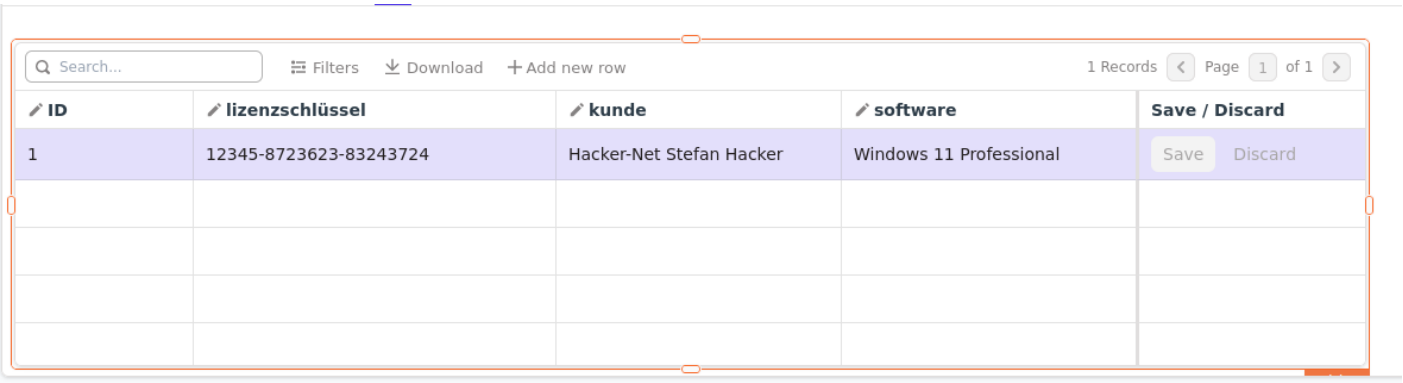
Nun werden die splaten dargestellt.
Dort den Haken rein bei Editable.
Nun kann man wenn man möchte auch den Speichern / verwefen Text ändern., wenn man möchte.
Dort kann man dann single row oder multirow update auswählen. Hier wählen wir erstmal single row.



Nun noch den Schieberegler bei bei Adding row rein.



Nun haben wir eine Tabelle mit der query.
Das widgets angepasst das man die Werte editieren kann (save / discard) und Einen neuen Datensatz hinzufügen kann.
Es fehlen noch die queries für die Aktionen.
Aber das widget ist somit eingerichtet



Editiervariante : 1

Die Verknüfpten Spalten sollen ein Dropdown Menü mit den Werten aus der jeweiligen Tabelle haben.
Bei Kunde wieder Firma,Vorname,Nachname und bei Software nur der Software Name.
Wenn in der Appsmith Widget Tabelle auf editieren geklickt wird soll der Wert aus der dem Dropdownmenü die ID eingesetzt werden.

Dazu erstellen wir uns erstmal zwei Queries für die Software Namen und die Kundennamen

Für Software

```
SELECT ID, name FROM software;
```

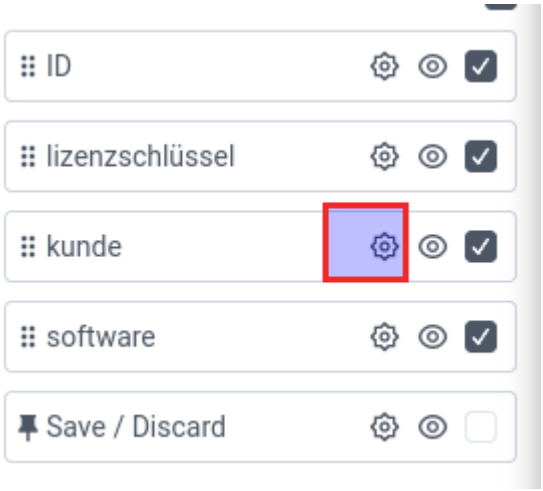
Diese speichern als select_software_dropdown

Für Kunden

```
SELECT ID,CONCAT(kunden.firma, '|' ,kunden.vorname, '|', kunden.nachname) AS kunde FROM kunden
```

Diese dann speichern als select_kunden_dropdown

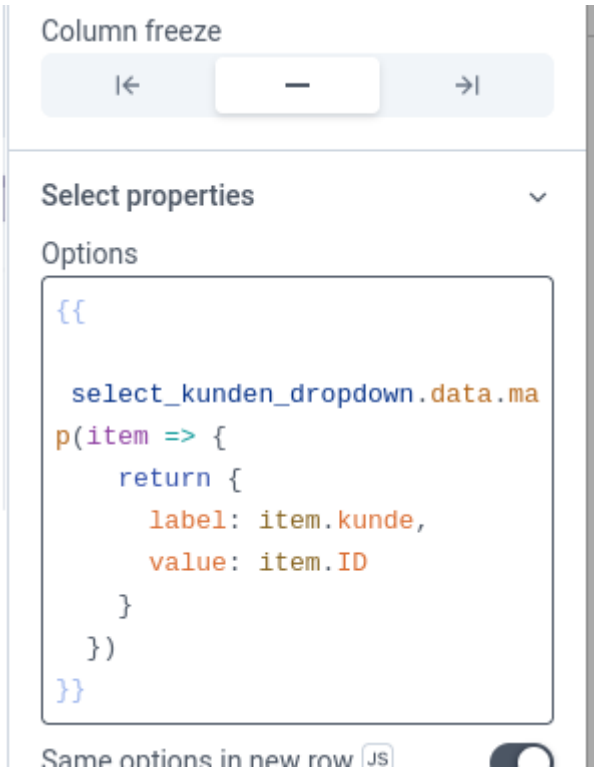
Wir ändern die Spalteneigenschaften in den Spalten kunde und software von Text auf Select.
Dazu auf das Zahnrad bei der Spalte klicken.



Nun den Coloumn Typ auf Select stellen

Jetzt unter select properties und dann options, folgendes eintragen.

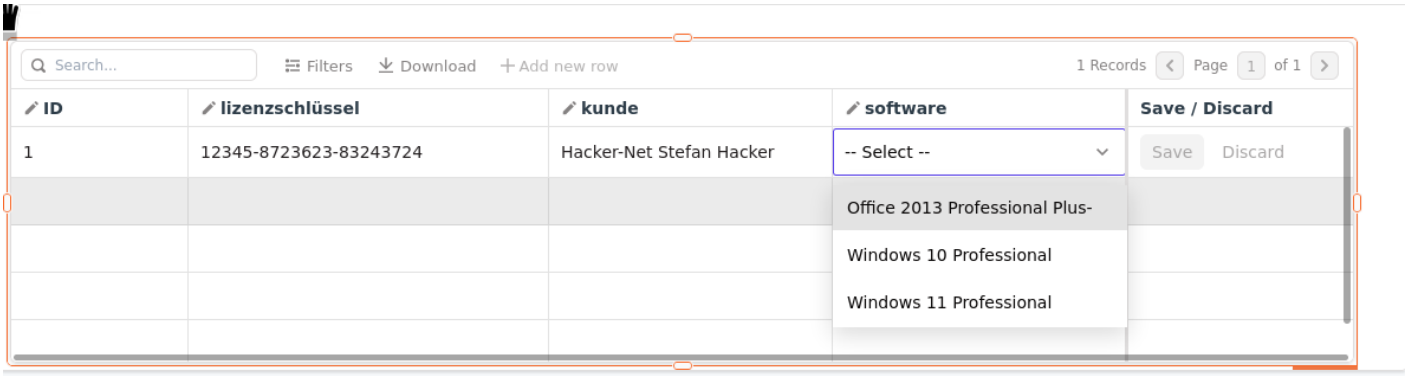
```
{{
  select_kunden_dropdown.data.map(item => {
    return {
      label: item.kunde,
      value: item.ID
    }
  })
}}
```



Das gleiche machen wir auch für die software

```
{{
 select_software_dropdown.data.map(item => {
  return {
   label: item.name,
   value: item.ID
  }
 })
}}
```

Nun haben wir wenn wir auf den Stift klicken eine Liste



Nun die Update query erstellen

UPDATE Lizenz SET Kunden_ID = @NeueKundenID, Software_ID = @NeueSoftwareID WHERE ID = 1 ; -- Hier sollte die spezifische Lizenz-ID stehen, die du aktualisieren möchtest