Ключевой момент.

Thoth не должен иметь привязки к графическому интерфейсу. Он должен иметь возможность управляться с консоли, десктопа или телефона.

Функционал.

1. Хранение.

Необходимо хранить информацию. Пользователь может создавать сколько угодно таблиц, но все они будут одинаковыми (при создании таблицы привязываются триггеры).

При добавлении записи в таблицу необходимо установить этой записи категорию.

При реализации записи так-же выбирается категория.

В системную таблицу-история (должна находится в той-же БД что и редактируемая таблица) заносятся все транзакции по добавлении/реализации.

1. Редактирование системных констант.

К системным константам относятся категории добавления и категории реализации, курсы валют, статус поступления?,

Примеры категории поступления: Продукты, Ремонт, Музыка, Техника

Примеры категории реализация: Мусор, Питание, Продажа

Системная таблица валют должна хранить курс по отношению к доллару.

1. Анализ
   1. Поступление (По категориям, по позициям, по дате)
   2. Реализация (По категориям, по позициям, по дате)

Должен быть учет ежемесячных трат таких как ЖКХ, бензин

При создании БД пользователь может выбрать один из используемых шаблонов настроенной БД. Если не выбран ни один из шаблонов, используется шаблон по умолчанию для создания системных таблиц.

В несконфигурированной БД содержатся только системные таблицы (см. в приложении на планшете с синей шапкой). Конфигурация БД происходит в SandBox.

БД можно сконфигурировать импортом шаблона.

SandBox (По функционалу схоже с 1С.Конфигуратор)

Возможности:

Создавать, редактировать системные константы ( Категория поступления, Категория реализации, Единицы измерения кол-ва, Валюта [при отключённой функции автоматического обновления])

Создание новых таблиц. Удаление, редактирование пользовательских таблиц.

Привязки внешних ключей в пользовательских таблицах

По сути действие конфигуратора сводится к набору SQL запросов. => ~~для шаблона можно использовать готовый файл с набором SQL запросов (SQL скрипт).~~

Для шаблона используются заготовленные jar-модули

Типы таблиц: TABLE, TABLE\_SYSTEM, TABLE\_SYSTEM\_INVINSIBLE.

TABLE – пользовательская таблица. Пользователь может создать, редактировать и удалить её через SandBox. Заполнение происходит через Viewer.

TABLE\_SYSTEM – системная таблица. Пользователь может создать, редактировать и удалить её через SandBox. Через Viewer происходит только чтение. Пример: Типы данных, Категория товаров, Валюта, Единицы измерения.

TABLE\_SYSTEM\_INVINSIBLE – системная таблица, недоступная пользователю даже через SandBox. К таким таблицам относятся: Описания таблиц, Список таблиц, Типы таблиц.

Работа подзапросов

Подзапросы для INSERT.

1. Проверяем TableColumn на наличие внешнего ключа.
2. Если внешний ключ есть выполняем подзапрос SELECT для определения ID строки внешней таблицы
3. Если внешнего ключа нет, вставляем значение ContentValues.

Подзапросы для SELECT

1. Проверяем TableColumn на наличие внешнего ключа
2. Определяем столбцы для отображения в запросе. Если внешний ключ есть – выполняем подзапрос SELECT для определения информации по ID строки внешней таблицы.
3. Если внешнего ключа нет, вставляем значение Columns

Шаблон подзапроса

select columnsName from `внешняя таблица` where

Логика работы внешнего ключа.

Если внешний ключ указывает на колонку, отличную от колонки “id” в Table description записываем какая колонка интересует текущую и используем её в подзапросах. При создании таблицы устанавливаем внешний ключ на идентификатор строки.

Если внешний ключ указывает только на таблицу, осуществляется указание внешнего ключа на столбец идентификатора строки внешней таблицы. В подзапросах работают все колонки кроме “id”.