Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Предполагается, что создаваемый сервис - дочерний для платформы ShariX.
Это значит, что он использует такую же основу структуры БД для простоты синхронизации данных.
Таким образом, конечный проект являет собой:

- Общую часть БД (синхронизируемая часть)
- Локальную часть БД (локальная часть, в данном случае - пример черновика ТЗ сервиса каршеринга)

Задание

На основе данных черновиков построить схему синхронизируемой части.
Сделать описание локальной части БД и API сервиса.

Общая информация

Описание структуры БД синхронизируемой части.
Она должна быть едина для всех видов сервисов. Не только для каршеринга. Вся специфика по каршерингу выносится отдельно в локальную часть.
В этой структуре мы говорим уже на абстрактном уровне (в скобках приведен смысл для каршеринга):

- пользователь (просто аккаунт в системе)
- провайдер услуги (ответственное лицо, которое предоставляет авто)
- ресурс (автомобиль)
- услуга (предоставление автомобиля)
- заказ (единица аренды автомобиля)
- взаимоотношения (наличие или отсутствие договорных отношений)

Имейте ввиду, что помимо этой части, есть еще локальные данные сервиса, а также БД самой платформы (ее часть описана в файле “Вспомогательные”)

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Синхронизируемая часть

Определения и вопросы:

Глобальный сервис - программный продукт, обеспечивающий возможность сделать заказ цепочки услуг из любых сервисов, пользуясь технологиями платформы ShariX, вне зависимости от точки входа заказа (через API).

is_global - означает необходимость синхронизации данных, так как данные должны храниться на основной платформе, чтобы их можно было использовать в других услугах и сервисах, реализованных на платформе.

is_visible - означает видимость для составления цепочки услуг. Цепочка услуг - последовательность услуг различных поставщиков/сервисов, покупаемых клиентом в “пакетном” режиме (например, аренда и доставка арендуемого предмета, или аренда нескольких предметов одновременно, или аренда и оказание услуги человеком на арендуемой площадке и так далее).

Meta-service - это видимая для клиента самостоятельная единица Сервиса как бизнеса. У одной компании (юрлица) может быть несколько сервисов. Один сервис может включать в себя результат взаимодействия нескольких компаний. Имеет свой брендинг, дизайн, позиционирование на рынке, набор конкретных услуг (service). Соответственно id_metaservice - идентификатор такого мета-сервиса.

id_metaservice - уникальный идентификатор мета-сервиса, необходимый для синхронизации данных. Один и тот же провайдер может быть для нескольких мета-сервисов, соответственно если происходят изменения в одном, то либо форсируется изменение во всех

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

(если возможно), либо снимается is_global. Соответственно при изменении is_global в true должно происходить согласование с остальными копиями в других сервисах. Нужен в том числе для того, чтобы выяснять, в каких еще сервисах есть этот провайдер.

service - идентификатор конкретной услуги. Услуга обладает ценой и исполнителем. Соответственно id_service - идентификатор такой услуги

datalink - адрес фактического размещения на физическом носителе, если информация настолько велика, что не может храниться внутри БД.

check-levels и check-level - это одно и то же, или разное? На данном этапе расцениваю это как одно и то же.

Клиентские аккаунты

Client — Клиент/пользователь/аккаунт в системе, который по логике получает услугу

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

userid	int	not null				id пользователя, которому соответствует роль клиента
requirements	name	not null				требования для того, чтобы можно было получать услуги как клиент
status	char	not null			может быть active только в том случае, если ticket, влияющий на статус - закрыт.	статус пользователя в системе относительно прохождения проверок (activity_status)
ticket_status	int				Если он меняет статус на закрытый - вызывается проверка, которая смотрит, нет ли другого открытого по пользователю.	id последнего актуального тикета, касающийся статуса.
id_metaservice	int	not null			При синхронизации используется для идентификации сервиса, в отношении которого устанавливаются настройки видимости.	уникальный идентификатор мета-сервиса.
is_visible	char	not null			не может быть невидим, но используемым для планирования.	по словарю **IS_GLOBAL/IS_VISIBLE** доступно для планирования в цепочке с другими услугами в

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

						глобальном сервисе
is_global	char	not null				по словарю **IS_GLOBAL/IS_VISIBLE** доступно для хранения в глобальном сервисе/необходима синхронизация

Примеры

Как оно сейчас в БД создано:

```
create table client (id serial primary key, userid int, requirements name, status char, ticket_status int, id_metaservice int, is_visible char, is_global char);
```

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла
`add_client_to_metaservice`	`userid, id_metaservice`				
`change_client_vis`	`userid,`				изменить

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

ibility	id_metaservice, is_visible				видимость конкретного клиента на указанную
change_client_globality	userid, id_metaservice, is_global				изменить способ хранения данных конкретного клиента на указанное
check_client_status				проверяется соответствие статуса и тикета, влияющего на статус	проверяет, может ли статус быть вообще назначен
find_client_tickets		список тикетов			выводится список тикетов, касающихся статуса и этого пользователя (может ли он его получить)

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Provider — единица описания поставщика услуг/ответственного лица за определенный ресурс (например, машину). По сути - это надстройка к клиентскому аккаунту, иллюстрирующая, что данный пользователь может выступать не только в роли потребителя. То есть, по тому, какие “провайдеры” находятся по идентификатору пользователя - можно установить конкретный список услуг данного пользователя.

id – уникальный идентификатор поставщика услуг/ответственного лица

type – тип поставщика (партнер/ответственное лицо/поставщик услуг). Смысл такой - провайдер это статус пользователя, который, в зависимости от применения, может нести разный смысл и подразумевает под собой какой-то тип действия. Обычные исполнители - это провайдеры услуг (код 3). Ответственные за какое-то имущество, которые сдают его в аренду - это тоже провайдеры (код 2). Ответственные за набор услуг перед метасервисом (фактически - назначенные админы) - это провайдеры-партнеры (код 1)

company (id) – уникальный идентификатор компании, от лица которой выступает провайдер. Смысл такой - ответственны могут быть только одушевленные лица, компании - не одушевленные. Все услуги предоставляются через компании-партнеры, самозанятые или ИП являются единицами таких компаний.

user_id (id) – уникальный идентификатор конкретного пользователя системы (meta-user), который будет оказывать услугу. Один пользователь может быть провайдером нескольких услуг. Статус провайдера означает, что с данным пользователем может быть установлена связь, как с исполнителем.

id_metaservice - уникальный идентификатор мета-сервиса, необходимый для синхронизации данных. Один и тот же провайдер может быть для нескольких мета-сервисов, соответственно если происходят изменения в одном, то либо форсируется изменение во всех (если возможно), либо снимается is_global. Соответственно при изменении is_global в true должно происходить согласование с остальными копиями в других сервисах. Нужен в том числе для того, чтобы выяснять, в каких еще сервисах есть этот провайдер.

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

requirements - требования для того, чтобы можно было предоставлять услуги любые в этом метасервисе в целом (самые строгие)

status - статус пользователя в системе относительно прохождения проверок (activity_status) (может быть active только в том случае, если ticket, влияющий на статус - закрыт.

ticket_status - id последнего актуального тикета, касающийся статуса. Если он меняет статус на закрытый - вызывается проверка, которая смотрит, нет ли другого открытого по пользователю.

is_visible (аккаунт поставщика услуг) – доступен для планирования в цепочке с другими услугами в глобальном сервисе

is_global (аккаунт поставщика услуг) – доступен для хранения в глобальном сервисе/необходима синхронизация

location_type - статическая или динамическая локация оказания услуги. Если статическая, а исполнитель находится существенно за пределами локации - то тогда статус автоматом оффлайн для приема новых заявок.

default_location - локация по умолчанию для объекта.

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Примеры

Как оно сейчас в БД создано:

```
create table provider (id serial primary key, type int, company int, userid int, id_metaservice int, requirements name, status char, ticket_status int, is_visible char, is_global char, location_type name, default_location name);
```

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла
add_provider				С provider от company обязательно должны быть у пользователя действующий relationship, иначе невозможно добавить строку.	

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

				С metaservice должно быть user_to_metaservice по локальным данным платформы. Связь с метасервисом - единственное исключение, когда отношение устанавливается не между парой пользователей.	

Resource/Список ресурсов – автомобили/дома/объекты сервиса

id - уникальный идентификатор ресурса

type_id - определение типа ресурса по его уникальному идентификатору в соответствии с классификатором

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

user_id (id_responsible_user?) – уникальный идентификатор ответственного (за состояние, доступность и так далее - то есть для договора) пользователя - идентификатор провайдера, по которому восстанавливается конкретный пользовательский аккаунт

requirements - код необходимого (самый строгий) для того, чтобы ресурс мог стать активным.

status - статус ресурса в системе относительно прохождения проверок (activity_status) (может быть active только в том случае, если ticket, влияющий на статус - закрыт.

ticket_status - id последнего актуального тикета, касающийся статуса. Если он меняет статус на закрытый - вызывается проверка, которая смотрит, нет ли другого открытого по пользователю.

is_global – доступны ли данные (по услугам или ресурсам?) для хранения в глобальном сервисе/необходима синхронизация

is_visible – доступно ли для планирования в цепочке с другими услугами в глобальном сервисе

id_metaservice - уникальный идентификатор мета-сервиса, необходимый для синхронизации данных. Один и тот же провайдер может быть для нескольких мета-сервисов, соответственно если происходят изменения в одном, то либо форсируется изменение во всех (если возможно), либо снимается is_global. Соответственно при изменении is_global в true должно происходить согласование с остальными копиями в других сервисах. Нужен в том числе для того, чтобы выяснять, в каких еще сервисах есть этот провайдер.

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary	serial	1		идентификатор записи

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

		key				

Примеры

Как оно сейчас в БД создано:

```
create table resources (id serial primary key, type_id int, user_id int, requirements name, status char,
ticket_status int, id_metaservice int, is_visible char, is_global char);
```

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Relationship/Взаимоотношение – описание связей (желательных - как имеющиеся договорные отношения, и нежелательных - как пожелание любой из сторон)

id - уникальный идентификатор связи

user_id_who (id) – уникальный идентификатор инициатора договорных отношений

user_id_whom (id) – уникальный идентификатор того с кем связываются

neg_type_id (id) - тип договорных отношений по его уникальному идентификатору

id_metaservice - уникальный идентификатор мета-сервиса, необходимый для синхронизации данных. Один и тот же провайдер может быть для нескольких мета-сервисов, соответственно если происходят изменения в одном, то либо форсируется изменение во всех

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

(если возможно), либо снимается is_global. Соответственно при изменении is_global в true должно происходить согласование с остальными копиями в других сервисах. Нужен в том числе для того, чтобы выяснять, в каких еще сервисах есть этот провайдер.

requirements - код необходимого (самый строгий) для того, чтобы ресурс мог стать активным. Оно вставляется автоматом, в соответствии с профилем метасервиса. Далее, если кому-то из партнеров или пользователей надо строже - применяется более строгий вариант на данную связь.

status - (статус обработки заявки в системе заявок)

ticket_status - id заявки, по которой происходит проверка статуса relationship. State меняется только в результате изменений в заявке.

is_global - установленный тип договорных отношений между клиентами/пользователями/аккаунтами доступен для хранения в глобальном сервисе/нужна синхронизация

is_visible - установленный тип договорных отношений между клиентами/пользователями/аккаунтами, доступен для планирования в цепочке с другими услугами в глобальном сервисе

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Примеры

Как оно сейчас в БД создано:

```
create table relationship (id serial primary key, user_id_who int, user_id_whom int, neg_type char, id_metaservice int, is_visible char, requirements name, status int, ticket_status int, is_global char);
```

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Company/Партнеры дочерних сервисов

id – уникальный идентификатор компании

legal_name – настоящее имя юридического лица

representative (provider_id) – уникальный идентификатор представителя компании. Это обязательно пользователь-провайдер определенного типа. То есть нельзя назначить ответственного, который не может быть ответственным.

inn (поле заполняется только после того, как загружен подтверждающий документ и проверен)
kpp (поле заполняется только после того, как загружен подтверждающий документ и проверен)
ogrn (поле заполняется только после того, как загружен подтверждающий документ и проверен)
bank_name - название банка с расчетным счетом
bik - бик
ks - корсчет
rs - расчетный счет
address - юрадрес

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

requirements - код необходимого (самый строгий) для того, чтобы ресурс мог стать активным. Оно вставляется автоматом, в соответствии с профилем метасервиса. Далее, если кому-то из партнеров или пользователей надо строже - применяется более строгий вариант на данную связь.

status - (статус обработки заявки в системе заявок)

ticket_status - id заявки, по которой происходит проверка статуса relationship. State меняется только в результате изменений в заявке.

id_metaservice - уникальный идентификатор мета-сервиса, необходимый для синхронизации данных. Один и тот же провайдер может быть для нескольких мета-сервисов, соответственно если происходят изменения в одном, то либо форсируется изменение во всех (если возможно), либо снимается is_global. Соответственно при изменении is_global в true должно происходить согласование с остальными копиями в других сервисах. Нужен в том числе для того, чтобы выяснять, в каких еще сервисах есть этот провайдер.

is_visible – доступно ли для планирования в цепочке с другими услугами в глобальном сервисе

is_global – доступно ли для хранения в глобальном сервисе/нужна синхронизация данных

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Примеры

Как оно сейчас в БД создано:

```
create table company (id serial primary key, legalname varchar(255), representative int, inn int, kpp int, ogrn int, bank_name int, bik int, ks int, rs int, address int, requirements name, status char, ticket_status int, id_metaservice int, is_visible char, is_global char);
```

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

При добавлении компании автоматом не создается relationship с representative и company, так как все relationship создаются между representative. С metaservice у него должна быть уже действующая связь.

servicetype/Перечень типов услуг

id - идентификатор типа услуги
codename - латинское наименование услуги в системе

caption - наименование услуги для отображения пользователю

description - текстовое описание услуги

requirements - код требований на основе вспомогательных таблиц-справочников

price_type - ценообразование - код допустимых вариантов или код параметров, принимаемых во внимание и способ их учета (по сути хорошо закодировать формулу)

status - активность на основе системы заявок

ticket_status - id последнего актуального тикета, касающийся статуса. Если он меняет статус на закрытый - вызывается проверка, которая смотрит, нет ли другого открытого по пользователю.

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

id_metaservice - уникальный идентификатор мета-сервиса, необходимый для синхронизации данных. Один и тот же провайдер может быть для нескольких мета-сервисов, соответственно если происходят изменения в одном, то либо форсируется изменение во всех (если возможно), либо снимается is_global. Соответственно при изменении is_global в true должно происходить согласование с остальными копиями в других сервисах. Нужен в том числе для того, чтобы выяснять, в каких еще сервисах есть этот провайдер.

is_visible – доступно ли для планирования в цепочке с другими услугами в глобальном сервисе

is_global – доступно ли для хранения в глобальном сервисе/нужна синхронизация данных

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи

Примеры

Как оно сейчас в БД создано:

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

```
create table servicetype (id serial primary key, codename varchar(255), caption varchar(255), description text, requirements name, pricetype int, status char, ticket_status int, id_metaservice int, is_visible char, is_global char);
```

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Service/Перечень уникальных услуг поставщиков

service - спецификация услуги каждого конкретного поставщика (например, в рамках сервиса многие могут предоставлять услуги перевозки, но конкретный шаблон с конкретным тарифом относится к отдельному перевозчику)

Id – уникальный идентификатор услуги

servicetype_id – тип оказываемой услуги по классификатору услуг сервиса

id_provider – уникальный идентификатор поставщика услуг (фактически определяет, какой пользователь будет оказывать услугу)

resource_id - так как ресурсы сами услугу оказать не могут, а также один ресурс может быть представлен в виде разных услуг, то фактически с точки зрения смысла системы ресурс - это как неодушевленный пользователь. Без провайдера, который с его помощью оказывает услугу - никуда. Поле остается пустым, если сервис не предусматривает использование услуг. Стоит обратить внимание, что это не обязательно ответственный за ресурс. Например, за состояние автомобиля может быть ответственен пользователь (он и указывается в таблице со свойствами ресурса), а услугу доступа или перевозки может оказывать иное лицо.

requirements - код необходимого (самый строгий) для того, чтобы ресурс мог стать активным. Оно вставляется автоматом, в соответствии с профилем метасервиса. Далее, если кому-то из партнеров или пользователей надо строже - применяется более строгий вариант на данную связь.

id_metaservice - уникальный идентификатор мета-сервиса, необходимый для синхронизации данных. Один и тот же провайдер может быть для нескольких мета-сервисов, соответственно если происходят изменения в одном, то либо форсируется изменение во всех

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

(если возможно), либо снимается is_global. Соответственно при изменении is_global в true должно происходить согласование с остальными копиями в других сервисах. Нужен в том числе для того, чтобы выяснять, в каких еще сервисах есть этот провайдер.

price_alg – шаблон алгоритма расчета цены для оказываемой услуги (по этой переменной определяется, какую функцию для расчета цены вызывать)

price_km - значение параметра стоимости 1км данного поставщика для данного шаблона услуги

price_min - значение параметра стоимости 1мин данного поставщика для данного шаблона услуги

price_amount - значение параметра стоимости 1 услуги данного поставщика для данного шаблона услуги

service_status - статус спецификации типа услуги, принимает значения Online, Offline, Preorder with Gap. Online/offline выставляются по проверке параметров и желанию пользователя (например, если пользователь переключает себя online, но по какой-то причине ему такую услугу оказывать запрещено - оно не переключится, то есть надо перед сменой значения этого поля всегда запускать проверку)

status - активность на основе системы заявок

ticket_status - id последнего актуального тикета, касающийся статуса. Если он меняет статус на закрытый - вызывается проверка, которая смотрит, нет ли другого открытого по пользователю.

is_global - доступно ли для хранения в глобальном сервисе/нужна синхронизация данных

is_visible - доступно ли для планирования в цепочке с другими услугами в глобальном сервисе

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи

Примеры

Как оно сейчас в БД создано:

```
create table service (id serial primary key, servicetype_id int, id_provider int, resource_id int, requirements name, id_metaservice int, price_alg char, price_km int, price_min int, price_amount int, service_status char, status char, ticket_status int, is_visible char, is_global char);
```

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

Documents - это одна таблица со всеми документами. Вообще в концепции предполагалось, что таких таблиц должно быть много под каждый тип для удобства поиска. То есть отдельно таблица с паспортами, отдельно с правами, отдельно с какими-нибудь разрешениями и так далее. Что пока непонятно - документов может быть много разных.

id - уникальный идентификатор документа

id_metaservice - уникальный идентификатор мета-сервиса, необходимый для синхронизации данных. Если при синхронизации возникает конфликт (несовместимость) с другим сервисом, предлагается или форсировать изменения везде (если возможно), либо is_global выставляется как false.

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

doc_type - тип документа (паспорт/паспорт 1 страница и т д) в соответствии с классификатором типов документов (см описание в Requirements)

expire_date - срок окончания действия документа.

datalink - адрес фактического размещения на физическом носителе, если информация настолько велика, что не может храниться внутри БД.

userid - уникальный идентификатор пользователя (конкретного клиентского аккаунта) являющегося владельцем данного документа

company_id - идентификатор компании, к которой относится документ, если таковая есть (может не быть)

is_global - доступны ли документы для хранения в глобальном сервисе/нужна синхронизация

is_visible - доступна ли информация о наличии документов для планирования в цепочке с другими услугами в глобальном сервисе

check_level (check-levels) - информация об уровне проверки. Документ может быть проверен как платформой, так и мета-сервисом, так и партнером мета-сервиса, а может быть и никем (просто загружен). Указывается, так как достоверность проверки разная. Документ, проверенный только на низком уровне, не принимается во внимание как имеющийся до прохождения более высокоуровневой проверки. Информацию об уровнях проверки можно посмотреть по словарю Requirements. В данной таблице хранится информация о наиболее высоком уровне проверки.

checked_by - userid проверившего

check_date - timestamp проверки

status - активность на основе системы заявок

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

ticket_status - id последнего актуального тикета, касающийся статуса. Если он меняет статус на закрытый - вызывается проверка, которая смотрит, нет ли другого открытого по пользователю.

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи

Примеры

Как оно сейчас в БД создано:

```
create table documents (id serial primary key, id_metaservice int, doctype varchar(2), datalink varchar(255), userid int, company_id int, is_visible char, is_global char, check_level char, checked_by int, check_date timestamptz, status char, ticket_status int);
```

Новая версия, как надо создавать:

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Permissions/Разрешения - (проверки/экзамены). По смыслу это что-то вроде “документа на право что-то делать” - на данном этапе это ограничено метасервисом/платформой, при этом он может быть полностью цифровым (выданным платформой/сервисом).

id - уникальный идентификатор разрешения

id_permission - уникальный идентификатор определяющий наличие разрешения из множества в словаре - выданных пользователю/клиенту/аккаунту

id_metaservice - уникальный идентификатор мета-сервиса, необходимый для синхронизации данных.

check_level - тип проверки в соответствии с классификатором проверок.

user_id - уникальный идентификатор пользователя/клиента/аккаунта, которым была пройдена проверка и получено разрешение

is_global - доступна ли информация для хранения в глобальном сервисе/нужна синхронизация

is_visible - доступна ли информация о наличии разрешения для планирования в цепочке с другими услугами в глобальном сервисе

checked_by (check-level из классификатора платформы) - информация об уровне проверки. Проверка может быть проведена как платформой, так и мета-сервисом, так и партнером мета-сервиса, а может быть и никем (просто загружен). Указывается, так как достоверность проверки разная. Экзамен, проверенный только на низком уровне, не принимается во внимание как имеющийся до прохождения более высокоуровневой проверки.

checked_by - userid проверившего

check_date - timestamp проверки

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

status - (статус обработки заявки в системе заявок)

ticket_status - id заявки, по которой происходит проверка статуса relationship. State меняется только в результате изменений в заявке.

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи

Примеры

Как оно сейчас в БД создано:

```
create table permissions (id serial primary key, id_metaservice int, id_permission name, user_id int, is_visible char, is_global char, check_level char, checked_by int, check_date timestamptz, status char, ticket_status int);
```

Новая версия, как надо создавать:

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

id заявки всегда хранится в глобальной форме - то есть id_metaservice-ticket_id, потому что номера заявок глобальны.

Orders/Заказы current_orders (смысл такой, что записи в этой таблице могут меняться по ходу осуществления заказов, после - заказ записывается в историю - order_list_log)

id - уникальный идентификатор заказа

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

service_id - спецификатор услуги провайдера, нужен для установления цены (id_service - уникальный идентификатор шаблона услуги, необходим для установления цены и исполнителей.)
service_type_id - тип заказа по классификатору услуг

state - текущий статус заказа из возможных на платформе

id_metaservice - уникальный идентификатор мета-сервиса, необходимый для синхронизации данных

time_placed - время размещения заказа

time_start - время начала оказания услуги

time_finish_predicted - предварительное/расчетное время до окончания оказания услуги

time_finish_real - фактическое время окончания (точное установленное время)

order_place_start - место, которое фиксируется как точка с которой начался совершаться заказ

order_place_predicted предполагаемое место (например - парковка) завершения заказа, может совпадать с начальным, а также обязательно должно быть доступным для завершения (в случае с авто - в разрешенной зоне завершения, например)

order_place_real - фактическое место где был завершен заказ (например - машина припаркована “здесь”)

provider (id) - уникальный идентификатор поставщика услуги/аккаунта, который оказывает услугу. Если несколько провайдеров собираются мета-сервисом в цепочку, где на уровне связи с клиентом нельзя установить одно ответственное лицо, то указывается вспомогательный мета-провайдер сервиса, и это означает, что мета-сервис несет ответственность перед пользователем за сборку услуги воедино.

receiver (id) - пользователь/аккаунт, который принимает оказываемые услуги

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

client (id) -клиент/аккаунт, который оплачивает все оказанные услуги

predicted_price - расчетная цена с учетом тарифа поставщика услуг

real_price - цена с учетом тарифа поставщика услуг по факту оказания услуги

is_global - доступна ли информация по заказу для хранения в глобальном сервисе/нужна синхронизация данных. Если is_global = false, то и is_visible для заказа и вглубь по цепочке для всех исполнителей и ресурсов - тоже false.

is_visible - доступна ли информация по заказу (время, место) для планирования иных цепочек. Если нет, то все действующие исполнители и ресурсы считаются занятыми на неопределенное время, пока не завершится заказ. Если да - то ресурсы могут использоваться для построения цепочек после планируемого времени завершения, с учетом места.

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Примеры

Как оно сейчас в БД создано:

```
create table current_orders (id serial primary key, service_id int, service_type_id int, state int, id_metaservice int, time_placed timestamptz, time_start timestamptz, time_finish_predicted timestamptz, time_finish_real timestamptz, provider int, receiver int, client int, predicted_price int, real_price int, is_visible char, is_global char);
```

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

predicted_price рассчитывается посредством вызова функции get_price на основе данных о поставщике, услуге, тарифе для услуги и предполагаемой длительности и километражу (если он актуален).

real_price рассчитывается после превышения predicted_price с интервалом раз в 10 минут и/или по завершению заказа

TODO

1. Перенести имеющуюся информацию в формальный шаблон описания (таблица + комментарии)
2. Дописать информацию по примерам данных
3. В соответствии с документацией уточнить типы данных https://postgrespro.ru/docs/postgresql/9.6/datatype
4. Если необходимо откорректировать информацию в примерах - сделать это
5. Дописать примеры создания записей в БД (для новых типов данных, просто строки и так далее)

Возможные профили в рамках сервиса

1 Служебный пользователь (это обычный пользователь, у которого есть permissions соответствующие уровню - возможно для оптимизации стоит их дублировать в отдельную таблицу, также возможно стоит туда добавлять информацию о штатных часах работы, когда пользователь должен бы быть по-хорошему активен)

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

2 Партнер - company
Профиль Requirements для Drive по умолчанию:

3 Исполнитель
Профиль Requirements для Drive по умолчанию:

4 Клиент
Профиль Requirements для Drive по умолчанию:

5 Автомобиль
Профиль Requirements для Drive по умолчанию:

6 Тип услуг
Профиль Requirements для Drive по умолчанию:

7 Взаимоотношения
Профиль Requirements для Drive по умолчанию:

Основные системы сервиса

Машина (может быть много, каждая зарегистрированная физическая машина в системе определяется как ресурс и обладает своим уникальным id)
Приложение пользователя
Веб-приложение
Система Drive (бэкэнд сервиса)
Система ShariX (бэкэнд платформы)

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Описание локальных таблиц

current_timetable

(таблица с предстоящими заказами на период времени вперед - для оптимизации - неплохо бы ее держать отсортированной по времени начала например где-то)
id
order_id (из current_orders)
service_type_id - идентификатор типа конкретной услуги
provider_id - исполнитель
resource_id - ресурс, если есть
time_start_predicted - ожидаемое время начала
time_finished_predicted - ожидаемое время окончания
gap - промежуток времени в минутах до начала заказа в зависимости от типа заказа, времени суток и иных правил, определяемых сервисом (в словаре сервиса правило определения)

Таблица полностью строится на имеющихся данных и несет исключительно вспомогательный характер!

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи
order_id	int	not null		1		идентификатор заказа
service_type_id	int	not null		1		идентификатор типа конкретной услуги

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

provider_id	int	not null		1		исполнитель
resource_id	int			1		ресурс
time_start_predicted	timestamptz	not null	timestamptz	2022-02-23 22:59:55.555882+03		ожидаемое время начала
time_finished_predicted	timestamptz		timestamptz	2022-02-23 22:59:55.555882+03		ожидаемое время окончания
gap	int		int	60		промежуток времени в минутах до начала заказа

Примеры

Как оно сейчас в БД создано:

```
create table current_timetable (id serial primary key, order_id int, service_type_id int, provider_id int, resource_id int, time_start_predicted timestamptz, time_finished_predicted timestamptz, gap int);
```

Новая версия, как надо создавать: -

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице: -

Заготовки для API

Какие функции надо выполнить для того, чтобы запись была создана: не актуально для данной таблицы, валидность данных проверяется в оригинальных таблицах.

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла
get_upcoming_orde					Получить

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

`rs`					информацию о всех заказах в таблице
`actuality_check`					Проверить все заказы в таблице - а должны ли они там отображаться
`insert_current_order`					Добавить заказ в список текущих
`del_current_order`	`order id`				Удалить заказ из списка текущих
`update_current_order`	`order id`, параметры для замены и их значения				Обновить информацию в списке текущих

Tickets

tickets - таблица для внутренней системы заявок для обеспечения работоспособности сервиса
id - id заявки
reporter - создатель или заявитель
assignee - на кого назначена
time_created - время создания
time_assigned - время назначения на ответственного
time_changed - время последнего изменения
time_closed - время закрытия (потери актуальности финальной)

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

status - статус обработки (в соответствии с типом)
description - описание заявки (для человека или для автоматической обработки)

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание
id	serial	primary key	serial	1		идентификатор записи
reporter	int	not null		1		создатель или заявитель
assignee	int			2		на кого назначена
time_created	`timestamptz`	not null		2022-02-23 22:59:55.555882+03		время создания
time_assigned	`timestamptz`			2022-02-23 22:59:55.555882+03	Не должно быть раньше времени создания	время назначения на ответственного
time_changed	`timestamptz`			2022-02-23 22:59:55.555882+03	Не должно быть раньше времени создания	время последнего изменения
time_closed	`timestamptz`			2022-02-23 22:59:55.555882+03	Не должно быть раньше времени создания	время закрытия
status	int	not null				статус обработки (в соответствии с типом)
description	text					описание заявки (для человека или для

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

						автоматической обработки)

Примеры

Как оно сейчас в БД создано:

```
create table tickets (id serial primary key, id_reporter int, id_asignee int, time_created timestamptz,
time_assigned timestamptz, time_changed timestamptz, time_closed timestamptz, status int, description text);
```

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

(Точно такая же таблица есть на основной платформе и она не синхронизируется)

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла
`create_ticket`	все данные				Тикеты можно только создавать, закрытые тикеты не удаляются
`update_ticket_description`				проверка возможности через `check_ticket_permissions`	

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

`assign_ticket`	`id, assignee`			надо проверить, возможно ли этот тикет так назначить по правам через `check_ticket_permissions`	
`change_ticket_status`	`id, status`			надо бы лог куда-то про это писать проверка возможности через `check_ticket_permissions`	
`check_ticket_permissions`					проверка на возможность внесения изменения в указанный тикет

order_list_log - список всех услуг, оказанных сервисом (по сути оплаченных транзакций - то есть то, что уже произошло)

id
payment_transaction_id - id транзакции
service_id - спецификатор услуги провайдера

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

service_type_id - тип заказа по классификатору услуг

time_placed - время размещения заказа

time_start - время начала оказания услуги

time_finish_predicted - предварительное/расчетное время до окончания оказания услуги

time_finish_real - фактическое время окончания (точное установленное время)

order_place_start - место, которое фиксируется как точка с которой начался совершаться заказ

order_place_predicted предполагаемое место (например - парковка) завершения заказа, может совпадать с начальным, а также обязательно должно быть доступным для завершения (в случае с авто - в разрешенной зоне завершения, например)

order_place_real - фактическое место где был завершен заказ (например - машина припаркована “здесь”)

price - фактически снятое по чеку за услугу

provider (id) - уникальный идентификатор поставщика услуги/аккаунта, который оказывает услугу. Если несколько провайдеров собираются мета-сервисом в цепочку, где на уровне связи с клиентом нельзя установить одно ответственное лицо, то указывается вспомогательный мета-провайдер сервиса, и это означает, что мета-сервис несет ответственность перед пользователем за сборку услуги воедино.

receiver (id) - пользователь/аккаунт, который принимает оказываемые услуги

client (id) -клиент/аккаунт, который оплачивает все оказанные услуги

Название	Тип данных	Primary	Более удачный	Пример	Особенности и	Описание

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

столбца	по описанию	key / not null / unique	тип данных в соответствии с документацией Postgres	данных	ограничения данных (длина, символы, откуда берутся и прочее)	
id	serial	primary key	serial	1		идентификатор записи

Примеры

Как оно сейчас в БД создано:

```
create table order_list_log (id serial primary key, payment_transaction_id int, service_id int, time_placed timestamptz, time_start timestamptz, time_finish_predicted timestamptz, time_finish_real timestamptz, order_place_start name, order_place_predicted name, order_place_real name, price int, provider int, receiver int, client int);
```

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Логика такая

Есть конечная транзакция - то, за что был выписан чек (платформой). Тут хранятся данные о том, за что чек.
service - абстракция на уровень выше. Это услуга какого-то провайдера, которую он может предоставлять. На этом этапе может вычисляться цена (по сути - это информация о том, что конкретный провайдер может брать деньги по конкретной схеме).
service_type - еще на уровень выше. Эта схема услуги, которую могут разные провайдеры применять в рамках сервиса. Шаблон услуги сервиса.

Таблицы для открывания/закрывания автомобиля

current_cars

resource_id - id машины, о которой речь
order_id - id заказа, по которому он забронирован (NULL если свободен)
user_id - id пользователя забронировавшего (для ускорения запросов) (NULL если свободен)
order_status - статус заказа (NULL если свободен)
current_token - токен контроллера, установленного в машине

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

doors_output - статус дверей
engine_output - статус блокировки двигателя
central_lock_input - статус центрального замка
ignition_input - зажигание
doors_input - статус дверей
input4 - неведомый input 4

Информация от контроллера, которую не обязательно логировать, но можно периодически опрашивать контроллер с целью мониторинга (раз в 30 сек?):

controller_status (online/offline)
signal_gsm - сила сотового сигнала
operator_gsm

signal_gps - сила сигнала gps
Location
Speed
Direction
Address

battery_level
board_voltage

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Примеры

Как оно сейчас в БД создано:

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

cars_log

resource_id
timestamp - время записи лога
operation - какая операция была произведена
user_id - пользователь, ответственный за инициализацию операции

Название столбца	Тип данных по описанию	Primary key / not null / unique	Более удачный тип данных в соответствии с документацией Postgres	Пример данных	Особенности и ограничения данных (длина, символы, откуда берутся и прочее)	Описание

Примеры

Как оно сейчас в БД создано:

Новая версия, как надо создавать:

Пример запроса на добавление записи для прототипа на основе данных в примере в таблице:

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Заготовки для API

Название функции	Вход	Выход	Запросы к БД	ТЗ к логике	Описание смысла

Словари

neg_type - типы взаимоотношений с пользователями

Диапазон:

Код	Словесное название	Описание
0	disfollow	пользователь выразил предпочтение не устанавливать

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

		взаимоотношений с другим пользователем (не включать его в цепочки услуг при наличии других исполнителей, при отсутствии - только в виде предположения)
1	prefer	пользователь зафиксировал предпочтение использования услуг другого пользователя при построении цепочек услуг. При наличии таких предпочтений пользователю обязательно предлагается одно из решений на основе данных связей, если это возможно. Остальные - предложениями.
2	employee	пользователь является сотрудником
3	service	Пользователь является сервисом (предоставляет сервис)
4	client	устанавливается связь для получения услуги

type_legal_form - классификация типов организационных форм

Код	Словесное название	Описание
0	self-employed	Самозанятый
1	ip	ИП
2	company	Организация

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

resource_type

Классификатор ресурсов (автомобили, дома, иные объекты)

Код	Словесное название	Описание
1 бит (движимость)		
0	moving	движимое
1	still	недвижимое
2 бит - среда размещения объекта во время оказания услуги		
0	ground	
1	underground	
2	water	
3	underwater	
4	air	
5	cosmos	
3 бит - класс имущества		
0	zero	совсем дно

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

1	economy	
2	comfort	
3	business	
4	premium	
5	luxe	
4-5 бит - тип имущества		
00	car	
01	flat	
02	house	
03	airplane	
	…	

permission_type

Код	Словесное название	Описание

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

1 бит (уровень применения)		
0	platform	платформа
1	metaservice	метасервис
2 бит (тип разрешения)		
0	admin	администратор
1	supervisor	модератор/супервайзор
2	support	поддержка
3	techsupport	техподдержка
4	user	пользователь

initiator_type - типы инициированных событий в зависимости от инициатора

Код	Словесное название	Описание
0	manual	вручную (например, на этапе разработки, тестирования, отладки)
1	self	Сам пользователь, от которого запрос
2.*	another_user	иной пользователь, после кода 2 записывается id

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

		пользователя
3	metaservice_event	событие внутри метасервиса, после кода 3 записывается id event
4	platform_event	событие платформы, после кода 4 записывается id event

event_type - типы событий

Код	Словесное название	Описание
1	add	
2	del	
3	change	

IS_GLOBAL/IS_VISIBLE статусы

Код	Словесное название	Описание
f	false	не синхронизируется/не видно
t	true	синхронизируется, все ок

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

d	disabled	было включено, но стало выключено и в процессе обработки
s		было выключено, но стало включено и в процессе обработки
e	error	ошибка - то есть должно работать, но почему-то не ок

Requirements

Это последовательность символов, определяющая, какие документы/проверки должны быть осуществлены для того, чтобы статус был активным.

Требования могут быть со стороны платформы (S), метасервиса (M), юрлица-исполнтеля (P) - это check levels.

Записывается в виде строки S….M…P….U…. - если какие-то разделы пустые, то они обозначаются таковыми, а не опускаются.

. - любое число

Пример для водителя Drive для получения услуги:

DS M01z07z10o11o P01y07y10m11m CS M P34y Be

Пример для водителя Assist

DS01z07z11z M01y07z09m11m P01x07y11x CS..z M31y P14 Bd

D - разделитель для обозначения документов

коды степени проверки

z - просто загружен

y - загружен и проверен на уровне требования

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

x - загружен и проверен на уровень выше (сервисом или платформой)

o - опционален, может отсутствовать

m - опционален и проверен на уровне требования (если проверен - то хранится)

n - опционален и проверен на уровень выше (если проверен - то хранится)

коды документов

01 - паспорт

02 - инн

03 - снилс

04 - свидетельство о регистрации компании

05 - система налогообложения

06 - документ, подтверждающий полномочия представлять компанию (доверенность, приказ)

07 - документы, подтверждающие право совершать действия (права, лицензии)

08 - документы, подтверждающие собственность

09 - документ об образовании

10 - медицинская книжка

11 - справка об отсутствии судимости

12 - договор (в том числе о трудоустройстве)

13 - фотография

99 - иное

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

C - разделитель

коды степени проверки

z - автоматическая проверка (проверка нижестоящих о наличии)

y - проверка человеком на уровне требования

x - проверка человеком на уровень выше (сервисом или платформой)

o - опционален, может отсутствовать

m - опционален и проверен на уровне требования

n - опционален и проверен на уровень выше

коды проверок

1 разряд: очно (1) /звонок (2) /загрузка результата (3) /заочно онлайн (4)

2 разряд: тест с выбором ответа (1)/произвольный ответ на заданный вопрос (2)/ свободная беседа (3)/ свободная практика (4)

B - bindings - наличие связей

d - наличие связи пользователь-компания

e - наличие связи пользователь-метасервис

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Pricetype

Код	Словесное название	Описание
1	time	учитывается только время
2	distance	учитывается только расстояние
3	instance	учитывается количество
4	tpd	time per distance - время за расстояние
5	ipd	instances per distance - штуки за расстояние
6	ipt	instances per time - штуки за время

Location_type

Код	Словесное название	Описание
0	static	Не должно меняться (только при редактировании свойств пользователем)

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

1	dynamic	Может меняться в результате опроса устройства

Type (тип поставщика)

Смысл такой - провайдер это статус пользователя, который, в зависимости от применения, может нести разный смысл и подразумевает под собой какой-то тип действия. Обычные исполнители - это провайдеры услуг (код 3). Ответственные за какое-то имущество, которые сдают его в аренду - это тоже провайдеры (код 2). Ответственные за набор услуг перед метасервисом (фактически - назначенные админы) - это провайдеры-партнеры (код 1)

Код	Словесное название	Описание
1		партнер
2		ответственное лицо
3		поставщик услуг

Service_status

Код	Словесное название	Описание
1	online	в сети и готов принимать заказы
2	offline	не в сети и не готов принимать заказы

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

3	gap	preorder with gap - предзаказ по времени

activity_status (статус активности по совокупности ситуации с документами) - или заменяется статусами ACCESS_REQUEST!

Код	Словесное название	Описание
0	active	
1	deactivated	
2	deleted	для обозначения пользователей, который пожелали удалиться полностью

transaction_type (тип транзакции)

, ее инициатор (эквайринг или вручную - и что именно - оплата, бронирование, возврат, расчет с агентом и так далее)

Код	Словесное название	Описание
0	active	
1	deactivated	

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

STATE (статусы заказов/заявок с кодами)

Смысл такой. 1 разряд определяет тип заявки. 2 - шаг исполнения. 3 - статус. При этом 0 - это черновой или сервисный статус, требующий реакции, 1, 2… - штатные по шагам исполнения; 9, 8… - негативные статусы.

Тип заявки - обработка ситуации ST_REQUEST

Требующие рассмотрения сотрудниками техподдержки:

- 111 NEW
- 110 REOPENED

Требующие обработки сотрудниками компании:

- 121 ASSIGNED
- 131 IN PROCESS

Требующие обработки инициатором заявки:

- 149 WONTFIX
- 141 DONE

Не требующие дальнейшей обработки:

- 159 DUPLICATE
- 151 CLOSED

Тип заявки - Бронирование SERVICE_REQUEST

- 210 TEMPLATE (заявка формируется в режиме реального времени, сохранение временной информации, если это нужно, пред-бронирование). Условие попадания - пользователь начинает заполнять заявку/сохраняет черновик заявки.
- 211 BOOKED (бронирование создано). Условие попадания - пользователь отправляет заявку. Создается вспомогательная заявка ответственному лицу на проверку (если проверка не происходит автоматом).
- 221 ACCEPTED (бронирование подтверждено системой или ответственным лицом) Условие попадания - ответственное лицо его подтвердило.

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

- 220 PENDING (требуется дополнительная информация от пользователя по бронированию для подтверждения) Условие попадания - ответственное лицо изменило статус на данный в результате проверки заявки.
- 229 DECLINED (бронирование отклонено) Условие попадания - ответственное лицо изменило статус на данный.
- 222 PRE-START (предстартовое состояние - например, срабатывает по таймеру или событию за некоторое время до наступления бронирования) Условие задается для каждого типа сервиса отдельно. Переход в это состояние только для ACCEPTED заявок.
- 231 PROCESS (заявка в активной стадии обработки, процесс выполняется). Условие перехода - начинающий действовать (подписанный) договор на оказание услуги.
- 238 PRE-FORCEMAJEUER (состояние пред-форсмажора, вызванное каким-то событием) Условие перехода - автоматом в результате обработки состояния системы или вручную сотрудником техподдержки, особенность - процессу назначается ответственный сотрудник техподдержки, который ведет ситуацию до закрытия заявки или перехода обратно в PROCESS
- 239 FORCEMAJEUER (состояние форс-мажора). Условие перехода - подтверждение сотрудником техподдержки данного состояния, обрабатывается совместно с назначенным сотрудником до статусов закрытия или до статусов нормального течения оказания услуги
- 241 DONE (обработка заявки завершена, услуга оказана). Условие перехода зависит от сервиса - либо исполнитель, либо клиент первично говорит об окончании услуги. Иным сторонам отправляется заявка на подтверждение оказания услуги.
- 249 CANCELLED (обработка заявки прервана после начала оказания услуги, услуга не оказана, оплата сторонам не производится или производится по штрафным правилам). Условие перехода - только через техподдержку или иначе в соответствии с настройками сервиса
- 251 CLOSED (обработка заявки завершена и закрыта всеми участниками). Условие перехода - все участники подтвердили факт успешного оказания услуги (либо оно проходит по таймауту) - после этого происходит оплата между сторонами.

Тип заявки - Запрос доступа ACCESS_REQUEST

- 320 PENDING (может сопровождаться только указанием типа прав на запрос)
- 321 ACCEPTED (переключается только из PENDING)
- 359 DECLINED (запрос доступа отправлен и отклонен по каким-то причинам)

Все права принадлежат ООО “ШЭРИКС”. Использование допустимо для выполнения практических учебных заданий. Предполагается публикация под свободной лицензией в ближайшее время. Актуальная информация о статусе материалов проекта на http://sharix-app.org. При отсутствии данной информации материалы можно использовать только в учебных целях.

Тип заявки - Обработка взаимосвязи. Заявки одноразовые и перестают быть активными после исполнения NEG_REQUEST

- 420 PENDING (запрос отправлен одним лицом в адрес другого и не подтвержден)
- 421 ACCEPTED (запрос отправлен одной стороной и подтвержден второй, для типа заявки отзыва взаимосвязи - подтверждение второй стороной не требуется)
- 459 DECLINED (запрос отправлен одной стороной и отклонен другой, если заявка была ACCEPTED - то изменение)