

## Проектирование и нормализация базы данных по доставкам

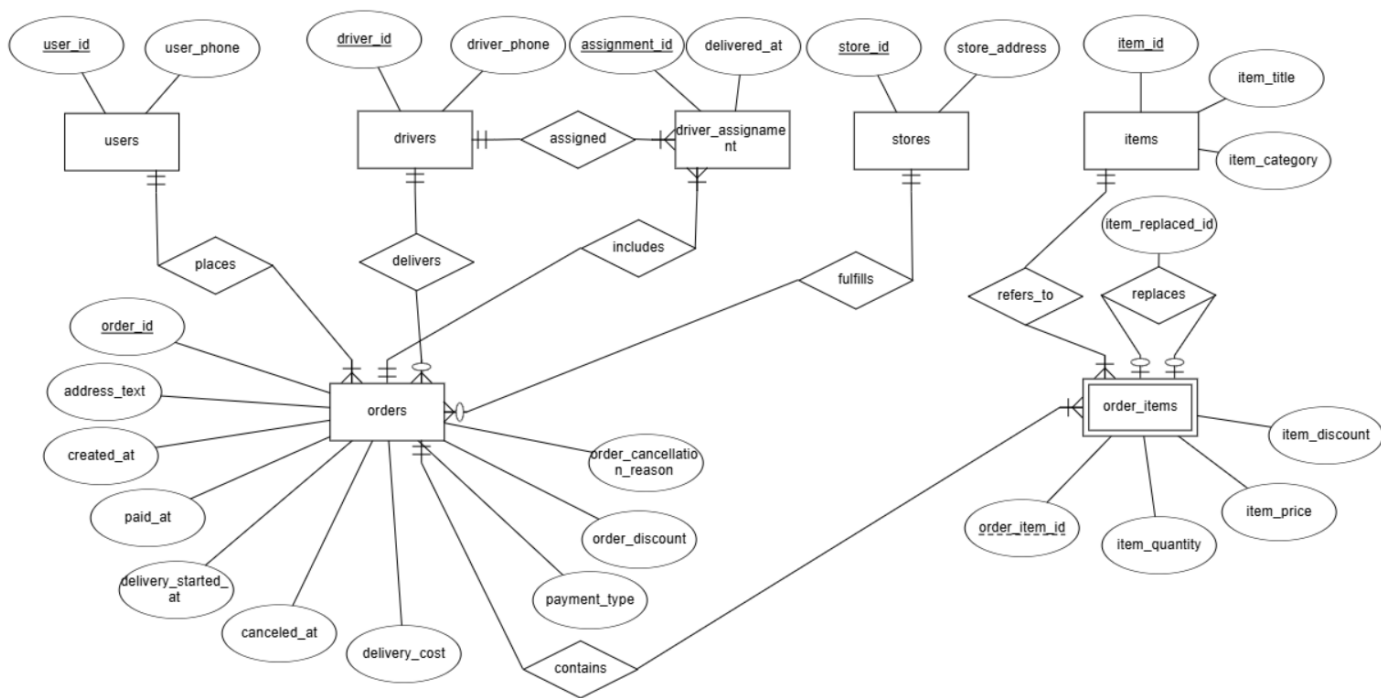
Представленные денормализованные данные содержали в себе фундаментальные структурные проблемы: каждая строка избыточно дублировала информацию о пользователях, водителях и магазинах, создавая риски аномалий при обновлениях, удалениях и вставках. Например, изменение номера телефона пользователя могло потребовать правки всех связанных записей, что неизбежно привело бы к рассинхронизации. Рекурсивная зависимость через `item_replaced_id` нарушает принцип атомарности, смешивая различные бизнес-события в одной структуре.

Для нормализации была выбрана третья нормальная форма (3NF), поскольку 3NF гарантирует, что все неключевые атрибуты зависят исключительно от первичного ключа, исключая транзитивные зависимости.

На первом этапе нормализации мы выделили основные бизнес-сущности: `users`, `drivers`, `stores`, `items` и `orders`, каждая со своими атомарными атрибутами. Также была создана слабая сущность `order_items`, которая разрешает составную природу заказов, выделяя информацию о товарах в отдельную структуру, зависимую от основного заказа. Дополнительно была введена отдельная ассоциативная сущность `driver_assignment`. Необходимость ее выделения обусловлена тем, что в рамках одного заказа может участвовать несколько курьеров, при этом только один из них фактически осуществляет доставку, тогда как для снятых с заказа курьеров атрибут `delivered_at` не заполняется. Данное поведение невозможно корректно отразить в рамках сущностей `orders` или `drivers` без нарушения нормальных форм.

Сущность `driver_assignment` связывает заказы и курьеров. В состав сущности был добавлен атрибут `assignment_id`, представляющий собой сгенерированный уникальный идентификатор. Этот атрибут отсутствовал в исходном наборе данных и был введен на этапе нормализации для однозначной идентификации факта назначения курьера на заказ и корректного хранения информации о времени доставки (`delivered_at`). Таким образом, каждый экземпляр сущности `driver_assignment` описывает отдельный факт участия конкретного курьера в конкретном заказе и связан внешними ключами с сущностями `orders` и `drivers`.

**В конечном итоге у нас получилась следующая концептуальная схема:**

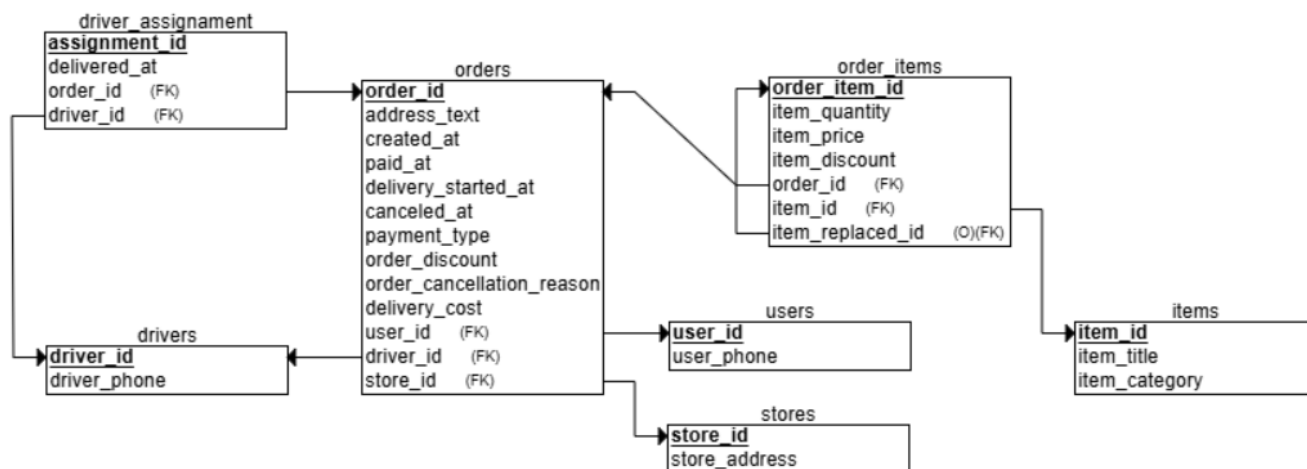


Изображение 1. ER-диаграмма.

## Обоснование связей:

Связь	Тип связи	Смысл	Ключевые поля
<b>users → orders</b>	Один-ко-многим (1:N)	Один пользователь может сделать один или много заказов Один заказ принадлежит только одному пользователю	users.user_id (PK) → orders.user_id (FK)
<b>drivers → orders</b>	Один-ко-многим (1:N)	Один водитель может выполнить 0, 1 или много доставок Один заказ доставляется только одним водителем	drivers.driver_id (PK) → orders.driver_id (FK)
<b>stores → orders</b>	Один-ко-многим (1:N)	Один магазин может выполнить 0, 1 или много заказов Один заказ собирается в одном магазине	stores.store_id (PK) → orders.store_id (FK)
<b>orders → order_items</b>	Один-ко-многим (1:N)	Один заказ содержит одну или много товарных позиций Одна позиция принадлежит только одному заказу	orders.order_id (PK) → order_items.order_id (FK)
<b>items → order_items</b>	Один-ко-многим (1:N)	Один товар из каталога может быть в одном или многих заказах Одна позиция в заказе ссылается на один товар из каталога	items.item_id (PK) → order_items.item_id (FK)
<b>order_items → order_items<sup>1</sup></b>	Рекурсивная один-к-одному (1:1)	Позиция заказа может заменять другую позицию, это позволяет отслеживать замену товаров внутри заказа	order_items.order_item_id (PK) → order_items.item_replaced_id (FK)
<b>orders → driver_assignments</b>	Один-ко-многим (1:N)	Один заказ может иметь несколько назначений курьеров Каждое назначение относится ровно к одному заказу	orders.order_id (PK) → driver_assignments.order_id (FK)
<b>drivers → driver_assignments</b>	Один-ко-многим (1:N)	Один курьер может быть назначен на множество заказов Каждое назначение относится ровно к одному курьеру	drivers.driver_id (PK) → driver_assignments.driver_id (FK)

Реляционная схема выглядит следующим образом:



Изображение 2. Реляционная схема.

<sup>1</sup> Замена товара моделируется самоссылкой в таблице order\_item. В заказе присутствуют обе записи: заменённый товар и товар-замена. У товара-замены заполнено поле item\_replaced\_id, указывающее на оригинальный товар.