Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Базы данных (БД)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовой работе
на тему:

«Онлайн-магазин автомобильных запчастей»

БГУИР КР 1-40 01 01 603 ПЗ

Студент: гр. 851006 Верещагин Н.В.

Руководитель: асс. Фадеева Е.Е.

Минск 2021

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ПОИТ

_____

(подпись)

Лапицкая Н.В. 2021 г.

ЗАДАНИЕ

по курсовому проектированию

Студенту Верещагин Николаю Владимировичу

1. Тема работы Онлайн-магазин автомобильных запчастей

2. Срок сдачи студентом законченной работы 20.12.2021 г.

3. Исходные данные к работе Документация по MySQL, Sparx Enterprise Architect

4. Содержание расчётно-пояснительной записки (перечень вопросов, которые подлежат разработке)

Введение
1 Анализ прототипов и литературных источников
2 Анализ требований и разработка функциональных требований
3 Инфологическая модель
4 Описание безнес-логики
5 Тестирование базы данных
Заключение, список литературы, приложения

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

1. "Онлайн-магазин автомобильных запчастей", А1, схема данных

6. Консультант по курсовой работе Фадеева Е.Е.

7. Дата выдачи задания 03.09.2021 г.

8. Календарный график работы над курсовой работой на весь период проектирования (с обозначением сроков выполнения и процентом от общего объёма работы):

раздел 1 к 15.09.2021 – 15 % готовности работы;

разделы 2, 3 к 15.10.2021 – 30 % готовности работы;

разделы 4, 5 к 15.11.2021 – 60 % готовности работы;

раздел 6 к 15.12.2021 – 90 % готовности работы;

оформление пояснительной записки и графического материала к 17.12.2021 – 100 % готовности работы.

Защита курсовой работы с 13 по 20 декабря 2021 г.

РУКОВОДИТЕЛЬ_____ Е.Е.Фадеева

(подпись)

Задание принял к исполнению_____ Н.В.Верещагин 03.09.2021 г.

(дата и подпись студента)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

С каждым годом интернет всё глубже проникает в деятельность мировых компаний, меняя стиль ведения бизнеса, его облик, предоставляя новые возможности и уникальные технологии для его развития. У интернета есть свои законы, свои возможности и особенности, свои преимущества и недостатки. Использование интернета в традиционном бизнесе приобретает стратегическое значение всех компаний. Это связано с экспоненциальным ростом количества посетителей интернета и с превращением его в основной канал продаж для многих отраслей бизнеса.

Выбранная тема считается актуальной на сегодняшний день, так как сегодня миллионы людей ежедневно, не выходя из дома, покупают различные товары в электронных магазинах. В мире, огромными темпами растет количество пользователей Internet и, как следствие, количество онлайн-покупателей.

Онлайн-магазины существенно уменьшают издержки производителя, сэкономив на содержании обычного магазина, расширяют рынки сбыта, так же, как и расширяет возможность покупателя – покупать любой товар в любое время в любой стране, в любом городе, в любое время суток, в любое время года. Это дает онлайн-магазинам неоспариваемое преимущество перед обычными магазинами. Этот момент является существенным при переходе производителей с «обычной» торговли на «онлайн».

Высокое качество продукции, умение донести информацию о продукте до потребителя и эффективная система сбыта, делает предприятие успешным на рынке. Во многих компаниях встречаются проблемы сбыта, которые мешают эффективно работать отделу продаж, и не исчезают даже с подбором хороших продавцов. Решить их можно только путем автоматизации процесса продаж. В узком и технологическом смысле, под онлайн-бизнесом ранее понималось использование информационных технологий (в первую очередь связанных с Интернетом) для организации взаимодействия предприятия с внешней средой, включая поставщиков, потребителей, партнеров и т.д. При таком подходе электронный бизнес выступает, прежде всего, как достаточно сложная прикладная информационная система. Более широкий, или концептуальный, подход рассматривает электронный бизнес как способ предпринимательства, способствующий достижению стратегического успеха в новую информационную эпоху. При таком понимании электронный бизнес отнюдь не сводится к информационным технологиям или активности в Интернете. Электронная коммерция затрагивает все аспекты бизнеса, включая стратегию, процессы, организацию и технологию, и выводит его далеко за сложившиеся границы. Online shop или e-shop — веб-сайт, рекламирующий товар или услугу, принимающий заказы на покупку, предлагающий пользователю выбор варианта расчета, способа получения заказа и выписывающий счет на оплату.

Онлайн-магазины стали важной частью нашей современной жизни, поэтому за последнее время можно заметить появления множества онлайн-магазинов. Для упрощения части их работы и достижения необходимой автоматизации им необходимо соответствующее программное обеспечение. В совокупности с возросшей потребностью в автомобильных запчастях, онлайн-магазины этой отрасли, пожалуй, достигли пика популярности. По этой причине выбор темы курсового проекта остановился на онлайн-магазине автомобильных запчастей.

1 АНАЛИЗ ПРОТОТИПОВ И ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ

1.1 Анализ литературных источников и прототипов

Реляционная база данных – это набор данных с предопределенными связями между ними. Эти данные организованны в виде набора таблиц, состоящих из столбцов и строк. В таблицах хранится информация об объектах, представленных в базе данных. В каждом столбце таблицы хранится определенный тип данных, в каждой ячейке – значение атрибута. Каждая строка таблицы представляет собой набор связанных значений, относящихся к одному объекту или сущности. Каждая строка в таблице может быть помечена уникальным идентификатором, называемым первичным ключом, а строки из нескольких таблиц могут быть связаны с помощью внешних ключей. К этим данным можно получить доступ многими способами, и при этом реорганизовывать таблицы БД не требуется.

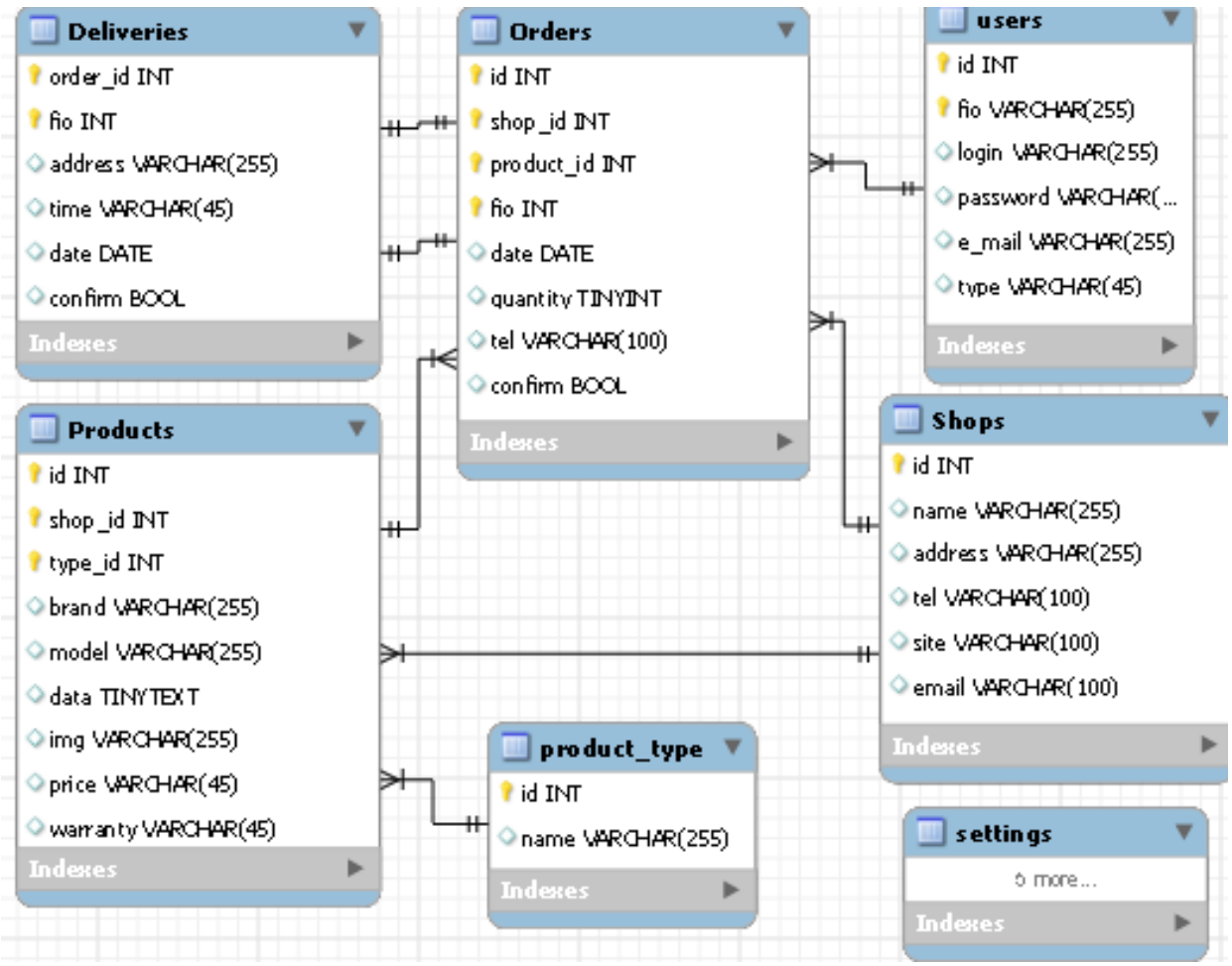


Рисунок 1.1 – Пример реляционной базы данных

1.2 Анализ существующих аналогов

1.1.1 Онлайн-магазин «TL24.BY»

TL24.BY – это белорусский интернет-магазин с ассортиментом товара, превышающим 150000 наименований. Ранее магазн занимался реализацией только автомобильных комплектующих. На данный момент онлайн-магазин занимается реализацией обширного списка продукции в том числе автомобильной. Благодоря организации магазинов-складов, TL24.BY предоставляет весьма приятные цены в сравнении с традиционными магазинами. Кроме того, он обладает довольно удобным сайтом, и развитой сетью пунктов самовывоза[3].

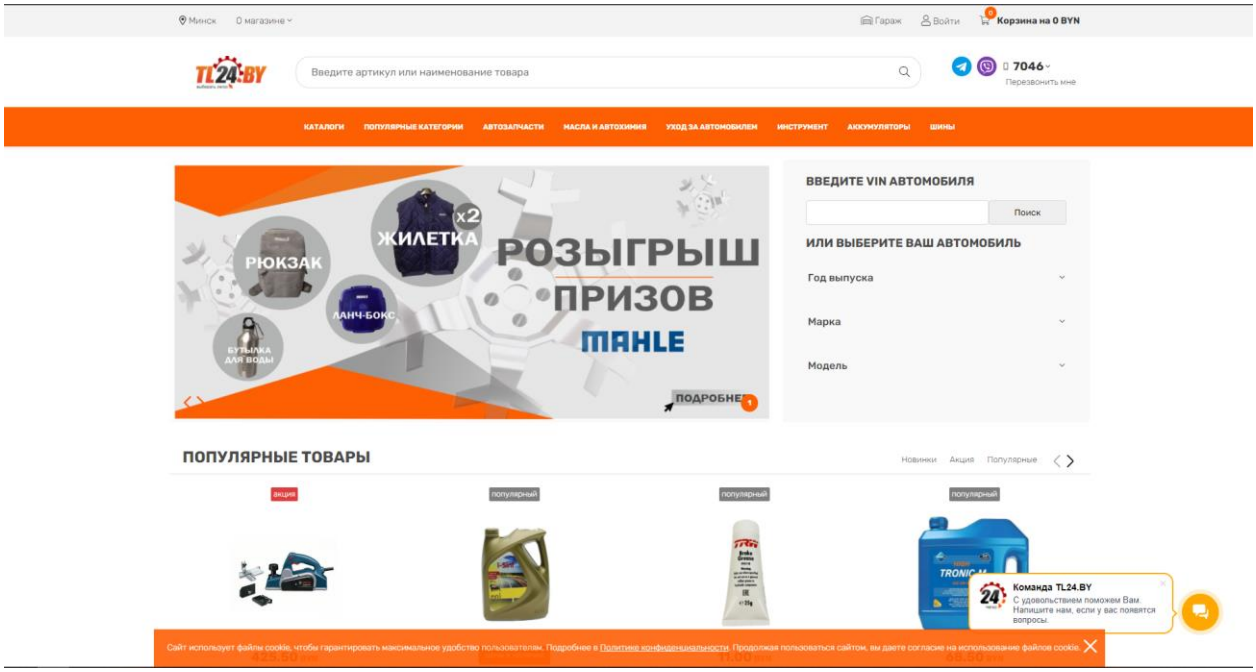


Рисунок 1.2 – Интерфейс «TL24.BY»

Достоинства:

– возможность взаимодействовать с программным обеспечением для чтения, печати и просмора информации;

– систематизирование данных;

– низкие цены на реализуемую продукцию;

– можество консультантов;

– наличие скидочных карт;

– возможность оплаты заказа онлайн.

Недостатки:

– устаревший интерфейс, который не предназначен в отличии болишнства сайтов конкурентов к мобильным устройствам;

– чрезмерная перегруженность интерфейса сайта;

– отсутствие оповещений клиента о появившихся интересующих его товаров.

1.1.2 Онлайн-магазин «DVMPARTS.BY»

DVMPARTS.BY – это белорусский интернет-магазин, который специализируется на продаже автомобилбных запчастей, а также техническом осмотре автомобилей. Данный магазин осуществляет доставку своих товаров по всей Беларуси, также предоставляет гарантию на все поставляемые им детали. Ассортимент меньше, чем у более крупных магазинов данной сферы[4].

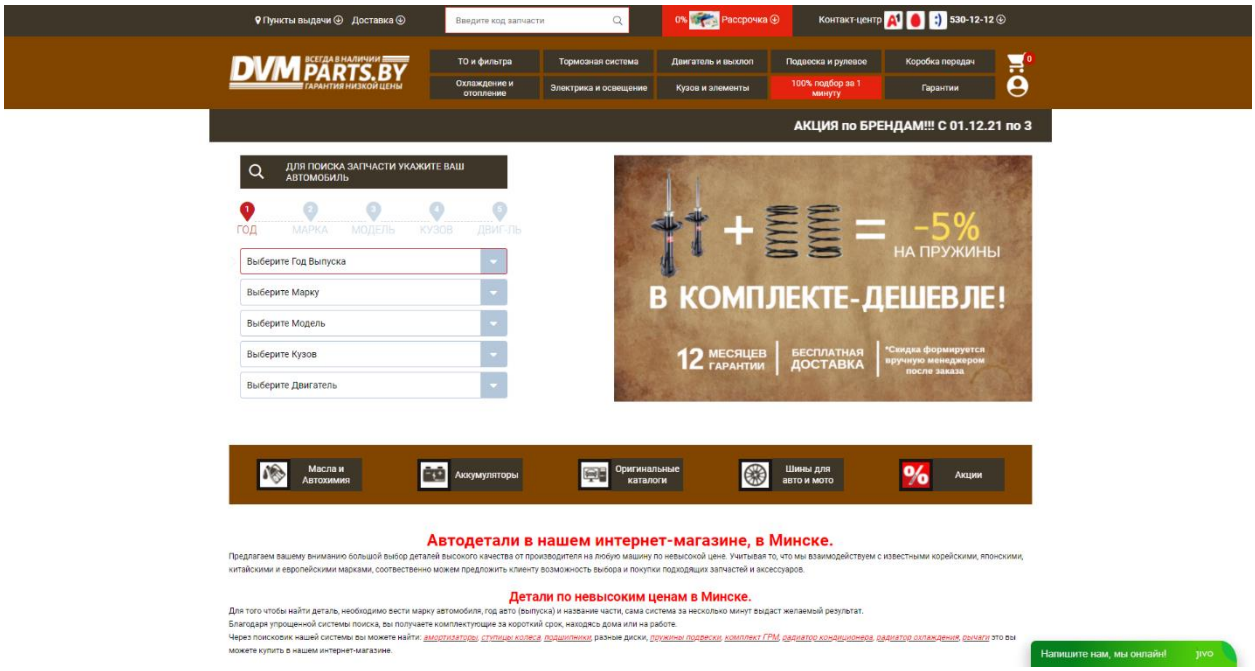


Рисунок 1.3 – Интерфейс «DVMPARTS.BY»

Достоинства:
– возможность взаимодействоввать с сайтом при промощи клавиатуры;
– систематизирование данных;
– низкие цены на реализуемую продукцию;
– множество консультантов;
– возможность создать список желаемого;
– возможность оплаты заказа онлайн.
Недостатки:
– чрезмерная перегруженность интерфейса сайта;
– малое количество пунктов самовывоза;
– низкая производительность сайта.

1.1.3 Онлайн-магазин «BYAVTO.BY»

Онлайн-магазин BYAVTO.BY работает на белорусском рынке уже 10 лет. Компания имеет многочисленные представительства в регионах Беларуси. Постоянно расширяющийся ассортимент каталога товаров, повышение качества услуг по продажам и доставке, делет покупки максимально удобными и, что немаловажно, выгодными[5].

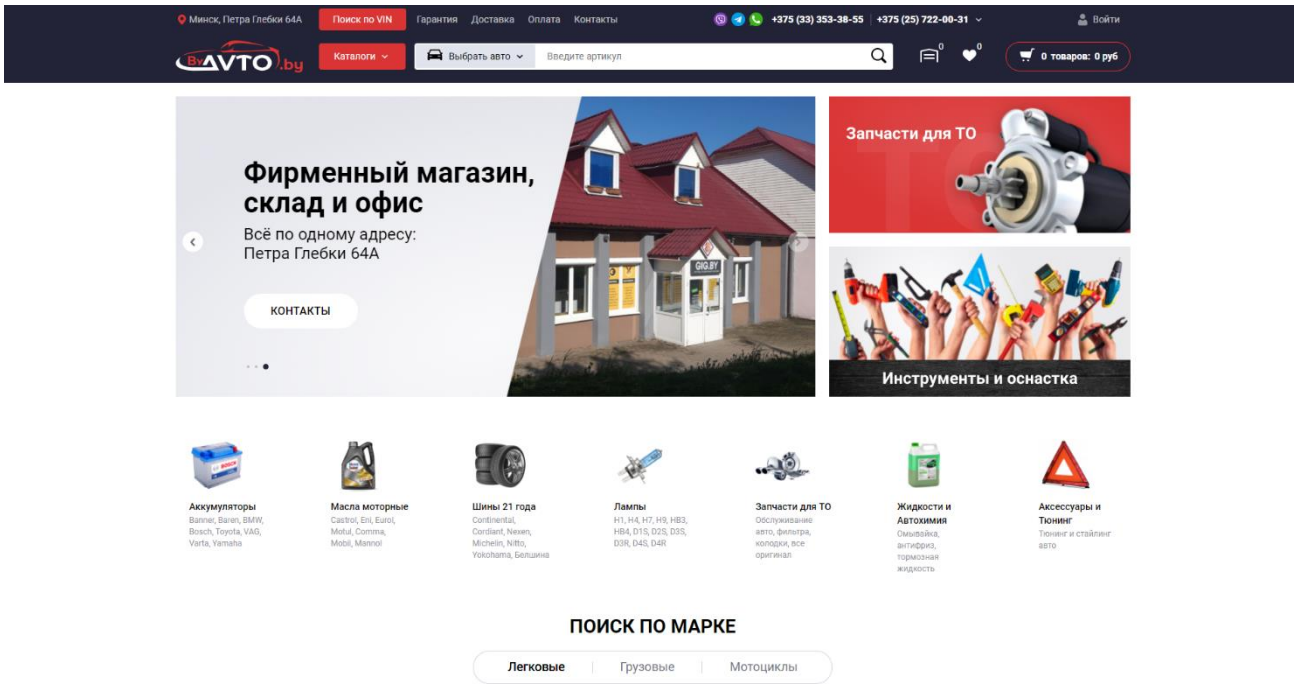


Рисунок 1.4 – Интерфейс «BYAVTO.BY»

Достоинства:
– широкий ассортимент реализуемой продукции;
– систематизирование данных;
– низкие цены на реализуемую продукцию;
– множество консультантов;
– возможность оставлять и просматривать комментарии к товарам;
– система оценивания товаров;
– стильный интерфейс, который подходит для любого устройства.
Недостатки:
– малое количество пунктов самовывоза;
– низкая производительность сайта;
– отсутствие возможности оплаты заказа онлайн.

1.3 Постановка задачи

На основании проанализированных схожих продуктов были выведены основные требования к проектируемой базе данных:

– база данных должна корректно обрабатывать входные данные;

– база данных должна предусмотреть добавление, обновление корректных данных и реакцию на триггеры добавления, обновления;

– база данных должна предусмотреть удаление данных и реакцию на триггеры удаления.

– база данных должна удалять данные в соответствии с запросами к БД;

– база данных должна обеспечить обобщение и систематизацию: автомобильных запчастей, заказов, доставок, для последующей эффективной работы над ними в системе;

Также для функционирования системы необходимо предусмотреть следующие элементы в базе данных:

– предоставление информации для каждого продукта (представление);

– предоставление информации о самых популярных продуктов по количеству заказов (прдставление);

– предоставление информации о самых эффективных доставщиках по количеству доставок (представление);

– предоставление информации о самых дорогих заказах (представление);

– предоставление информации о пользователях (представление);

– предоставление информации о заказах (представление);

– получение ID характеристики продукта (функция);

– получение значения храктер характеристики продукта (функция);

– создание заказа (функция);

– получение времени проведенным пользователем на сайте (функция);

– создание пользователя (функция);

– обновление информации о пользователе (процедура);

– очистка статистики времени проведенным на сайте пользователями (процедура);

– установка времени создания пользователя (триггер);

– установка времени создания заказа (триггер);

– очистка статистики времени проведенным на сайте пользователями при достижении максимальной велечины записей (триггер).

Для полноты представления системы необходимо учитывать существующие характеристики, типы автомобильных запчастей, а также к каким автомобилям относятся те или инные запчасти.

В качестве языка базы данных была выбрана СУБД MySQL, поскольку она сочетает в себе широкие возможности, и простоту написания реляционных баз данных. Необходимо, чтобы база данных работала в СУБД MySQL 8.027+.

2 АНАЛИЗ ТРЕБОВАНИЙ И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ

2.1 Требования к оборудованию

В качестве СУБД была выбрана MySQL (рисунок 2.1) т.к. она имеет необходимую простоту использования, наглядность, гибкость, низкую стоимость владения (относительно платных СУБД), а также масштабируемость и производительность.

MySQL позволяет хранить целочисленные значения со знаком и беззнаковые, длиной в 1, 2, 3, 4 и 8 байтов, работает со строковыми и текстовыми данными фиксированной и переменной длины, позволяет осуществлять SQL-команды SELECT, DELETE, INSERT, REPLACE и UPDATE, обеспечивает полную поддержку операторов и функций в SELECT- и WHERE- частях запросов, работает с GROUP BY и ORDER BY, поддерживает групповые функции COUNT(), AVG(), STD(), SUM(), MAX() и MIN(), позволяет использовать JOIN в запросах, в т.ч. LEFT OUTER JOIN и RIGHT OUTER JOIN, поддерживает репликацию, транзакции, работу с внешними ключами и каскадные изменения на их основе, а также обеспечивает многие другие функциональные возможности[7].

Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Есть и другие типы таблиц, разработанные сообществом.

СУБД MySQL появилась в 1995. Написана на C и C++, протестирована на множестве различных компиляторов и работает на различных платформах. С 2010 года разработку и поддержку MySQL осуществляет корпорация Oracle. Продукт распространяется как под GNU GPL, так и под собственной коммерческой лицензией. Однако по условиям GPL, если какая-либо программа включает исходные коды MySQL, то и эта программа тоже должна распространяться по лицензии GPL. Для нежелающих открывать исходные тексты своих программ как раз предусмотрена коммерческая лицензия, которая, в дополнение к возможности разработки под «закрытой» лицензией, обеспечивает качественную сервисную поддержку. Сообществом разработчиков MySQL созданы различные ответвления — Drizzle, OurDelta, Percona Server и MariaDB, все эти ответвления уже существовали на момент получения прав на MySQL корпорацией Oracle[2].

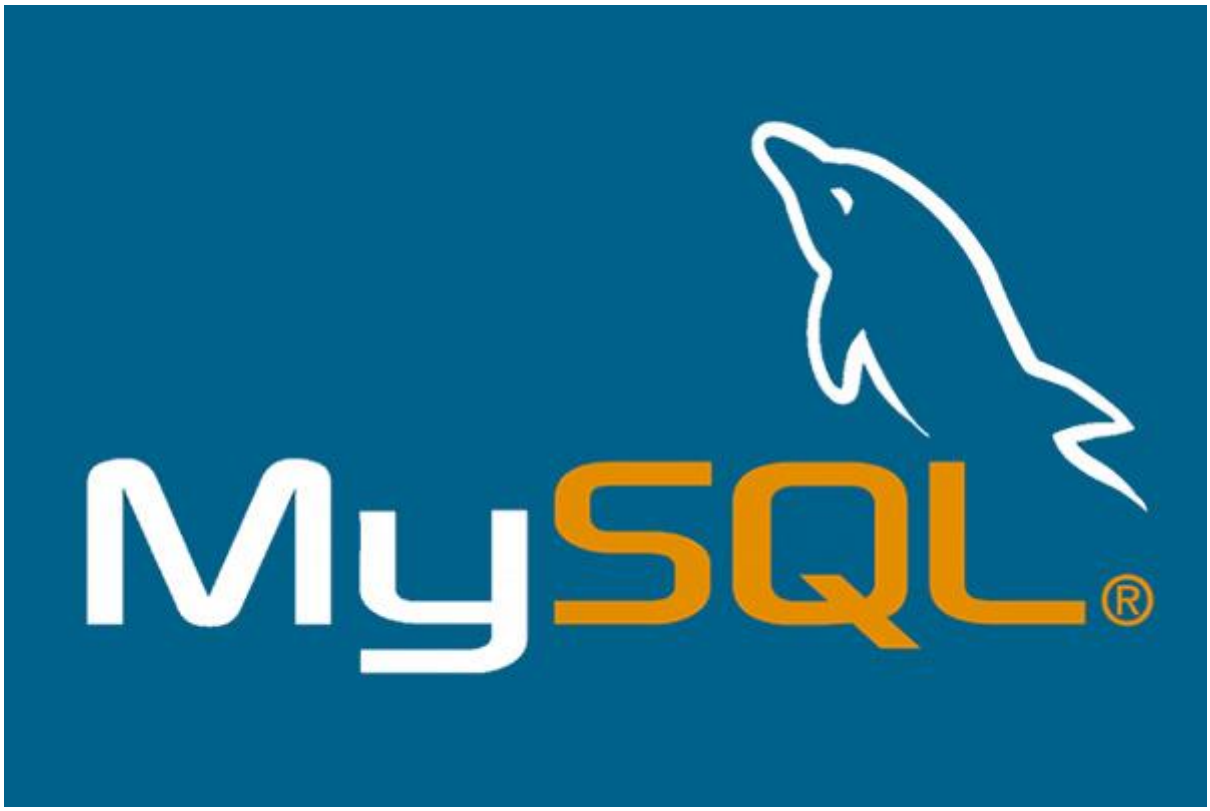


Рисунок 2.1 – Логотип MySQL

В качестве среды проектирования баз данных, была выбрана Sparx Enterprise Architect (рисунок 2.2), поскольку она предоставляет удобный и мощный инструментарий для проектирования моделей баз-данных разных уровней (от логического до визического). Кроме того, предоставляется инструментарий кодогенерации, что значительно ускоряет разработку и снижает вероятность ошибок.

Sparx Enterprise Architect - это инструмент моделирования полного жизненного цикла на основе UML, который используется для планирования, проектирования и создания программно-интенсивных систем и бизнес-процессов. Разработанный Sparx Systems, австралийской компанией-разработчиком программного обеспечения, основанной Джеффри Спарксом в 1996 году, Enterprise Architect доступен в четырех различных редакциях (вводная профессиональная версия, корпоративная командная версия, многофункциональная унифицированная версия и, наконец, версия Ultimate), каждая настроены для различных сценариев использования.

На данный момент у Enterprise Architect более 850,000 XNUMX пользователей по всему миру. Его пользователи охватывают широкий спектр отраслей, включая аэрокосмическую и оборонную, автомобильную, банковскую и финансовую, электротехническую, медицинскую, исследовательскую и академическую сферы, розничную торговлю, транспорт и коммунальные услуги. С момента своего первого выпуска Enterprise Architect стал предпочтительным инструментом моделирования UML для

разработчиков, консультантов и аналитиков, которые используют его не только для моделирования архитектуры своих систем, но и для обработки реализации этих моделей на протяжении всего жизненного цикла разработки приложений.

Набор инструментов Sparx Enterprise Architect разработан на базе UML 2.4.1, что помогает создать простые и легко обслуживаемые системы[1].

Он обладает рядом преимуществ:

– Скорость — загружает огромные модели за секунды;

– Стабильность и исполнительность;

– Программа обеспечивает доступ огромному количеству пользователей, которым доступен один и тот же вид на предприятие, как на локализованной территории, так и по всему Земному шару;

− Позволяет создавать динамические модели симуляции предприятий;

− Предоставляет возможность полного слежения за процессами: от требований, анализа и дизайна модели до реализации и развертывания;

− Позволяет группе лиц управлять комплексной информацией;

− Поддерживает много популярных языков, таких как C++, Visual Basic, PHP, Delphi и многие другие. Это позволяет сгенерировать или обратно спроектировать исходный код;

− Позволяет визуализировать Ваши приложения;

− Позволяет смоделировать базы данных, а также бизнес процессы;

Рисунок 2.2 – Логотип Sparx Enterprise Architect

2.2 Пользователи системы и их роли

В базе данных существуют следующие типы пользователей:

1. Системный администратор – пользователи, обеспечивающие функционирование системы, в том числе обслуживание БД (СУБД).

2. Администратор процесса – пользователи, осуществляющие настройку системы и регулирование прав доступа пользователей к функциональным подсистемам. Основная задача администратора процессов заключается в обеспечении работы остальных пользователей системы путем предоставления им прав доступа и распределения ролей. Администратор процессов также выполняет операции по контролю и исправлению ошибок в работе пользователей, мониторингу системы, заполнению, дополнению и изменению содержимого справочников системы.

3. Директор – сотрудник, который имеет полный доступ к системе. Также он может назначать права роли сотрудникам.

4. Доставщик – сотрудники, непосредственно осуществляющие доставку заказов на дом или пункты самовывозов с помощью системы.

5. Менеджер – сотрудники, которые: управляют заказами, назначают автомобили доставщикам, управляют продуктами, а также автомобилями и скидками, управляют точками самовывоза.

6. Покупатель – пользователь системы, который может сформировать заказ.

7. Управляющий складом – сотрудник, который управляет складом.

8. Управляющий пунктом самовывоза – сотрудник, который управляет пунктом самовывоза.

Основной поиск информации будет осуществляться с помощью запросов в базу данных по определенным критериям товаров. На стороне пользовательского интерфейса ожидается выбор интересуемых типов товара и их критериев. Затем серверная сторона обращается в базу данных, получая список товаров, соответствующих заданным критериям. После чего они должны быть отображены пользователю.

Значительная часть статистики будет браться из программного средства 1С, для упрощения работы с финансовыми операциями, и предотвращения необходимости получения лишних лицензий на подобные операции. Операции по начислению заработных плат, оплате покупок, оплате аренды, оплате налогов также будут проводиться с помощью 1С.

3 МОДЕЛЬ ПРЕДМЕТНОЙ ОБЛАСТИ

3.1 Сущности и связи

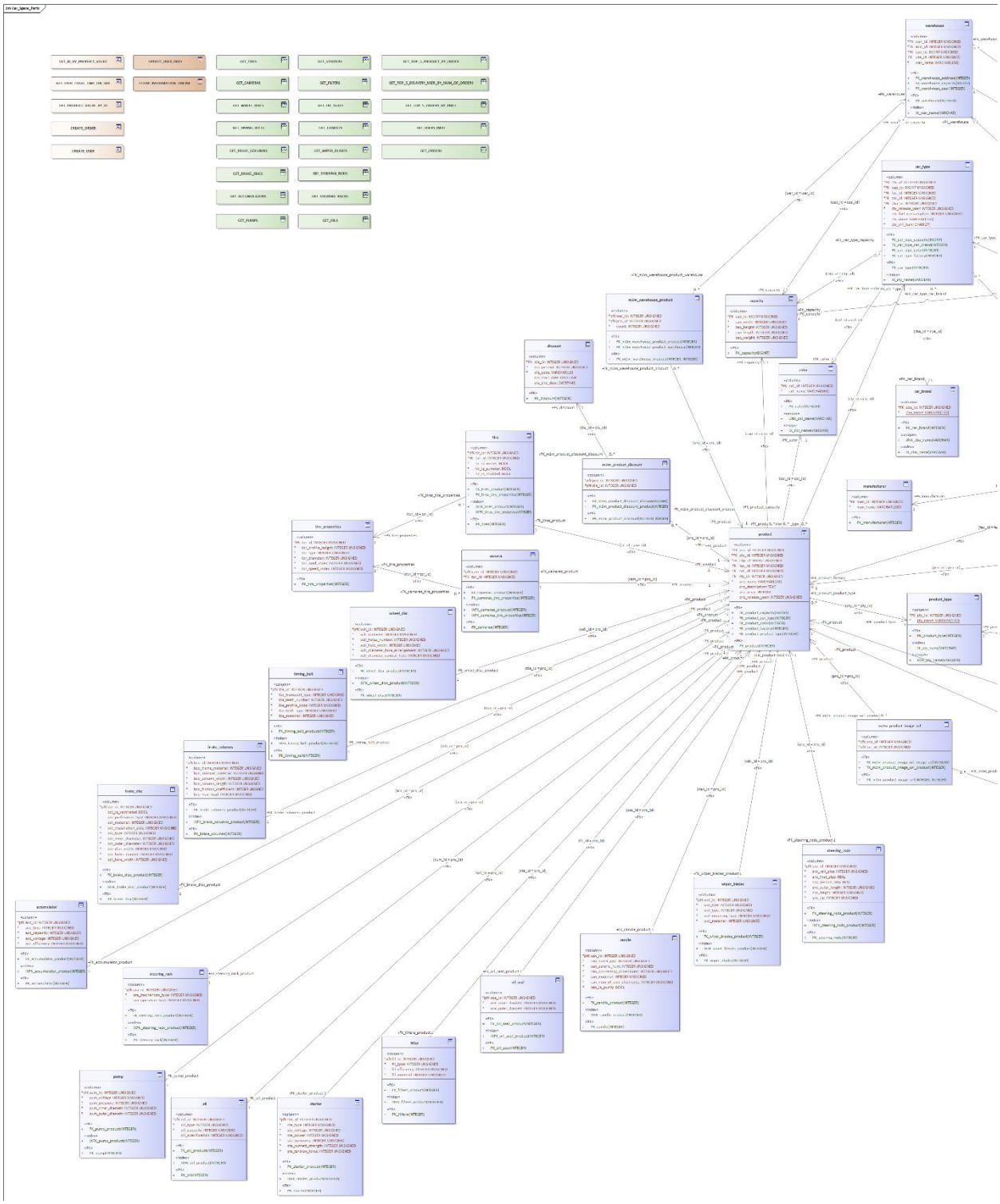


Рисунок 3.1 – 1 часть схемы базы данных

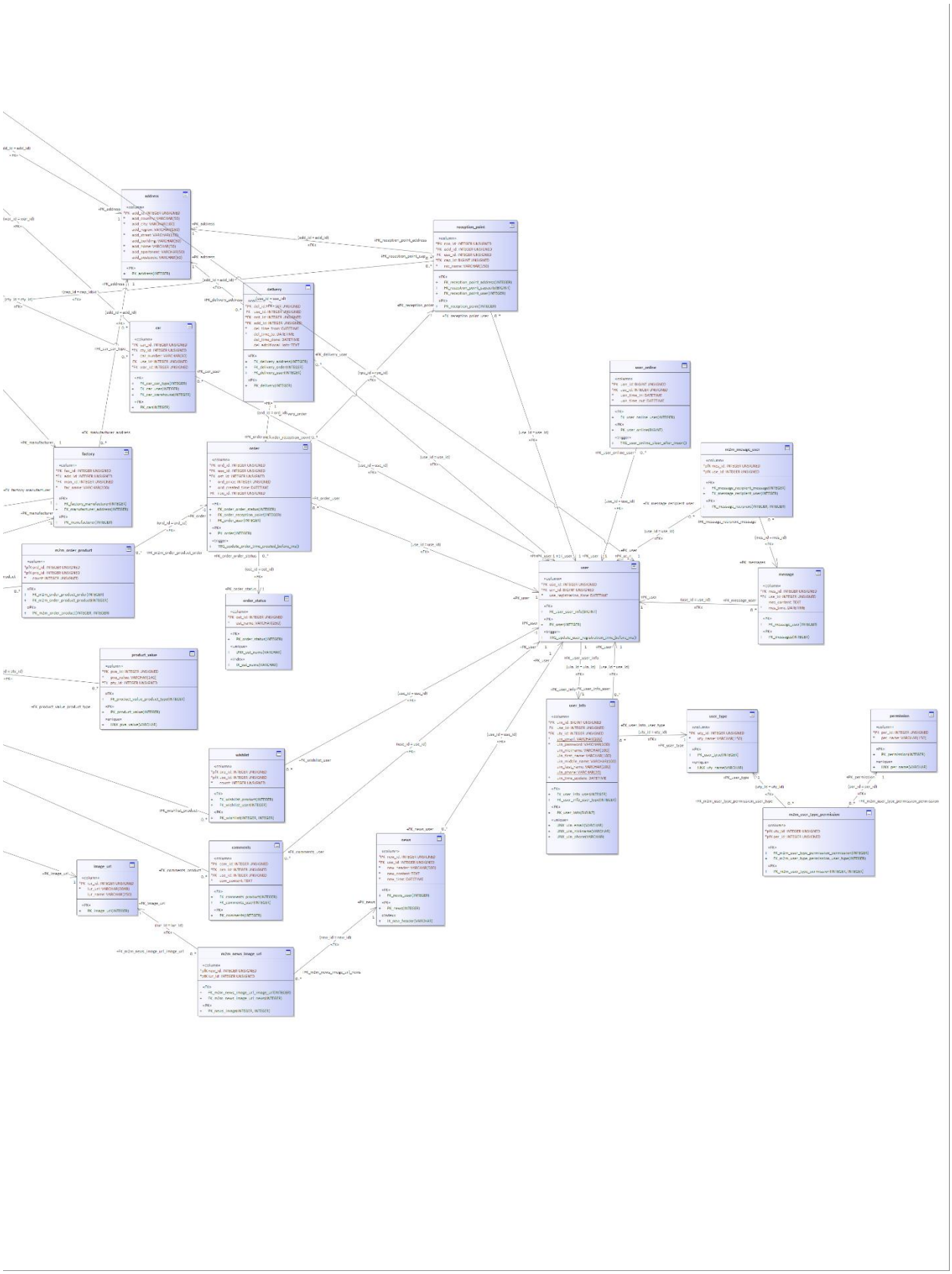


Рисунок 3.2 – 2 часть схемы базы данных

3.2 Особенности нормализации

Таблица product может содержать очень много узкоспециализированных полей, вследствие чего было принято решение вынести все эти поля в узкоспециализированные таблицы, уникальность достигается засчет применения связи один к одному. Уникальность многих полей (полей типов) достигается за счет получения их из таблицы product_value, где каждому id соответствует уникальный тип.

Таблица user_info была нормализована таким образом, поскольку пользователь может довольно часто изменять информацию о своих личных данных, а таблица user является одной из ключевых.

Таблица message была нормализована так, поскольку пользователи могут добавлять или создавать группы для общения.

Таблица car_type была нормализована данным образом, поскольку у многих автомобилей есть схожие характеристики (цвет, марка, производитель, вместимость).

Таблица capacity нормализована данным образом, потому что она используется не только для определения размерности товара, но также и для определения автомобильной вместимости и вместимости складов.

Таблица order нормализована так, поскольку не каждый заказ отправляется на пункт самовывоза.

Специализированные таблицы товаров в большинстве случаев имеют значительное количество полей, и при расширении функционала, их стоит вынести в отдельную таблицу.

4 СХЕМА БАЗЫ ДАННЫХ

```
/* ---------------------------------------------------------- */
/*  Generated by Enterprise Architect Version 15.0 */
/*  Created On : 17-Dec-2021 12:51:58 AM           */
/*  DBMS       : MySql                             */
/* ---------------------------------------------------------- */

SET FOREIGN_KEY_CHECKS=0
;
/* Drop Views */

DROP VIEW IF EXISTS `GET_ACCUMULATORS` CASCADE
;

DROP VIEW IF EXISTS `GET_BRAKE_COLUMNS` CASCADE
;

DROP VIEW IF EXISTS `GET_BRAKE_DISCS` CASCADE
;

DROP VIEW IF EXISTS `GET_CAMERAS` CASCADE
;

DROP VIEW IF EXISTS `GET_CANDLES` CASCADE
;

DROP VIEW IF EXISTS `GET_FILTERS` CASCADE
;

DROP VIEW IF EXISTS `GET_OIL_SEALS` CASCADE
;

DROP VIEW IF EXISTS `GET_OILS` CASCADE
;

DROP VIEW IF EXISTS `GET_ORDERS` CASCADE
;

DROP VIEW IF EXISTS `GET_PUMPS` CASCADE
;

DROP VIEW IF EXISTS `GET_STARTERS` CASCADE
;

DROP VIEW IF EXISTS `GET_STEERING_RACKS` CASCADE
;

DROP VIEW IF EXISTS `GET_STEERING_RODS` CASCADE
;

DROP VIEW IF EXISTS `GET_TIMING_BELTS` CASCADE
;

DROP VIEW IF EXISTS `GET_TIRES` CASCADE
;

DROP VIEW IF EXISTS `GET_TOP_5_DELIVERY_USER_BY_NUM_OF_ORDERS` CASCADE
;

DROP VIEW IF EXISTS `GET_TOP_5_ORDERS_BY_PRICE` CASCADE
;

DROP VIEW IF EXISTS `GET_TOP_5_PRODUCT_BY_ORDER` CASCADE
;

DROP VIEW IF EXISTS `GET_USERS_INFO` CASCADE
;
```

```
DROP VIEW IF EXISTS `GET_WHEEL_DISCS` CASCADE
;

DROP VIEW IF EXISTS `GET_WIPER_BLADES` CASCADE
;

/* Drop Tables */

DROP TABLE IF EXISTS `accumulator` CASCADE
;

DROP TABLE IF EXISTS `address` CASCADE
;

DROP TABLE IF EXISTS `brake_columns` CASCADE
;

DROP TABLE IF EXISTS `brake_disc` CASCADE
;

DROP TABLE IF EXISTS `camera` CASCADE
;

DROP TABLE IF EXISTS `candle` CASCADE
;

DROP TABLE IF EXISTS `capacity` CASCADE
;

DROP TABLE IF EXISTS `car` CASCADE
;

DROP TABLE IF EXISTS `car_brand` CASCADE
;

DROP TABLE IF EXISTS `car_type` CASCADE
;

DROP TABLE IF EXISTS `color` CASCADE
;

DROP TABLE IF EXISTS `comments` CASCADE
;

DROP TABLE IF EXISTS `delivery` CASCADE
;

DROP TABLE IF EXISTS `discount` CASCADE
;

DROP TABLE IF EXISTS `factory` CASCADE
;

DROP TABLE IF EXISTS `filter` CASCADE
;

DROP TABLE IF EXISTS `image_url` CASCADE
;

DROP TABLE IF EXISTS `m2m_mesage_user` CASCADE
;

DROP TABLE IF EXISTS `m2m_news_image_url` CASCADE
;

DROP TABLE IF EXISTS `m2m_order_product` CASCADE
;
```

```
DROP TABLE IF EXISTS `m2m_product_discount` CASCADE
;

DROP TABLE IF EXISTS `m2m_product_image_url` CASCADE
;

DROP TABLE IF EXISTS `m2m_user_type_permission` CASCADE
;

DROP TABLE IF EXISTS `m2m_warehouse_product` CASCADE
;

DROP TABLE IF EXISTS `manufacturer` CASCADE
;

DROP TABLE IF EXISTS `message` CASCADE
;

DROP TABLE IF EXISTS `news` CASCADE
;

DROP TABLE IF EXISTS `oil` CASCADE
;

DROP TABLE IF EXISTS `oil_seal` CASCADE
;

DROP TABLE IF EXISTS `order` CASCADE
;

DROP TABLE IF EXISTS `order_status` CASCADE
;

DROP TABLE IF EXISTS `permission` CASCADE
;

DROP TABLE IF EXISTS `product` CASCADE
;

DROP TABLE IF EXISTS `product_type` CASCADE
;

DROP TABLE IF EXISTS `product_value` CASCADE
;

DROP TABLE IF EXISTS `pump` CASCADE
;

DROP TABLE IF EXISTS `reception_point` CASCADE
;

DROP TABLE IF EXISTS `starter` CASCADE
;

DROP TABLE IF EXISTS `steering_rack` CASCADE
;

DROP TABLE IF EXISTS `steering_rods` CASCADE
;

DROP TABLE IF EXISTS `timing_belt` CASCADE
;

DROP TABLE IF EXISTS `tire` CASCADE
;

DROP TABLE IF EXISTS `tire_properties` CASCADE
;
```

```
DROP TABLE IF EXISTS `user` CASCADE
;

DROP TABLE IF EXISTS `user_info` CASCADE
;

DROP TABLE IF EXISTS `user_online` CASCADE
;

DROP TABLE IF EXISTS `user_type` CASCADE
;

DROP TABLE IF EXISTS `warehouse` CASCADE
;

DROP TABLE IF EXISTS `wheel_disc` CASCADE
;

DROP TABLE IF EXISTS `wiper_blades` CASCADE
;

DROP TABLE IF EXISTS `wishlist` CASCADE
;

/* Drop Stored Procedures */

DROP PROCEDURE IF EXISTS `CLEAR_INFORMATION_ONLINE`
;

DROP PROCEDURE IF EXISTS `UPDATE_USER_INFO`
;

/* Drop Functions */

DROP FUNCTION IF EXISTS `CREATE_ORDER`
;

DROP FUNCTION IF EXISTS `CREATE_USER`
;

DROP FUNCTION IF EXISTS `GET_ID_BY_PRODUCT_VALUE`
;

DROP FUNCTION IF EXISTS `GET_PRODUCT_VALUE_BY_ID`
;

DROP FUNCTION IF EXISTS `GET_USER_TOTAL_TIME_ON_SITE`
;

/* Create Functions */

DELIMITER //
CREATE FUNCTION CREATE_ORDER(
   `user` INTEGER UNSIGNED,
   `delivery_user` INTEGER UNSIGNED,
   `status` INTEGER UNSIGNED,
            `price` INTEGER UNSIGNED,
   `address` INTEGER UNSIGNED,
   `time_from` DATETIME,
   `time_to` DATETIME,
   `additional_info` TEXT
)
RETURNS INTEGER UNSIGNED DETERMINISTIC
BEGIN
            INSERT INTO `order`
   VALUES(NULL, `user`, `status`, `price`, CURDATE());

            INSERT INTO `delivery`
   VALUES(NULL, `user`, LAST_INSERT_ID(), `address`, `time_from`, `time_to`, NULL, `additional_info`);
```

```
    RETURN LAST_INSERT_ID();
END;
//
DELIMITER ;

;

DELIMITER //
CREATE FUNCTION CREATE_USER(
        `user_type` INTEGER UNSIGNED,
    `email` VARCHAR(100),
    `password` VARCHAR(100),
    `nickname` VARCHAR(100),
    `first_name` VARCHAR(100),
    `middle_name` VARCHAR(100),
    `last_name` VARCHAR(100),
    `phone` VARCHAR(20)
)
RETURNS INTEGER UNSIGNED DETERMINISTIC
BEGIN
        INSERT INTO `user_info`
    VALUES(
        NULL,
        NULL,
        `user_type`,
        `email`,
        `password`,
        `nickname`,
        `first_name`,
        `middle_name`,
        `last_name`,
        `phone`,
        CURDATE()
    );

    SET @user_info_id = LAST_INSERT_ID();
    INSERT INTO `user` VALUES(NULL, user_info_id, CURDATE());
    UPDATE `user_info` SET `use_id` = LAST_INSERT_ID() WHERE `uin_id` = @user_info_id;

    RETURN LAST_INSERT_ID();
END;
//
DELIMITER ;

;

DELIMITER //
CREATE FUNCTION GET_ID_BY_PRODUCT_VALUE(`val` VARCHAR(140), `prod_type` INTEGER
UNSIGNED)
RETURNS INTEGER UNSIGNED DETERMINISTIC
        RETURN (
                SELECT `pva_id`
                FROM `product_value` `pv`
                WHERE `val` = `pv`.`pva_value` AND `prod_type` = `pv`.`pty_id`
    );
//
DELIMITER ;

;

DELIMITER //
CREATE FUNCTION GET_PRODUCT_VALUE_BY_ID(`product_val_id` INTEGER UNSIGNED)
RETURNS VARCHAR(140) DETERMINISTIC
        RETURN (
                SELECT `pva_value`
                FROM `product_value`
                WHERE `product_val_id` = `pva_id`
    );
```

```
//
DELIMITER ;

;

DELIMITER //
CREATE FUNCTION GET_USER_TOTAL_TIME_ON_SITE(`user_id` INTEGER UNSIGNED)
RETURNS INTEGER UNSIGNED DETERMINISTIC
	RETURN (
		SELECT
			IFNULL(
				SUM(
					TIME_TO_SEC(`uon_time_out`) -
					TIME_TO_SEC(`uon_time_in`)
				), 0
			)
		FROM `user_online`
		WHERE `use_id` = `user_id`
   );
//
DELIMITER ;

;

/* Create Tables */

CREATE TABLE `accumulator`
(
	`acc_id` INT UNSIGNED NOT NULL,
	`acc_type` INT UNSIGNED NOT NULL,
	`acc_capacity` INT UNSIGNED NOT NULL,
	`acc_voltage` INT UNSIGNED NOT NULL,
	`acc_efficiency` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_accumulator` PRIMARY KEY (`acc_id` ASC)
)

;

CREATE TABLE `address`
(
	`add_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`add_country` VARCHAR(50) NOT NULL,
	`add_city` VARCHAR(100) NOT NULL,
	`add_region` VARCHAR(150) NULL,
	`add_street` VARCHAR(150) NOT NULL,
	`add_building` VARCHAR(50) NULL,
	`add_home` VARCHAR(50) NOT NULL,
	`add_apartment` VARCHAR(50) NOT NULL,
	`add_postcode` VARCHAR(50) NULL,
	CONSTRAINT `PK_address` PRIMARY KEY (`add_id` ASC)
)

;

CREATE TABLE `brake_columns`
(
	`bco_id` INT UNSIGNED NOT NULL,
	`bco_frame_material` INT UNSIGNED NOT NULL,
	`bco_internal_material` INT UNSIGNED NOT NULL,
	`bco_column_width` INT UNSIGNED NOT NULL,
	`bco_column_length` INT UNSIGNED NOT NULL,
	`bco_friction_coefficient` INT UNSIGNED NOT NULL,
	`bco_max_load` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_brake_columns` PRIMARY KEY (`bco_id` ASC)
)

;

CREATE TABLE `brake_disc`
```

```
(
	`bdi_id` INT UNSIGNED NOT NULL,
	`bdi_is_ventilated` BOOL NULL,
	`bdi_perforation_type` INT UNSIGNED NULL,
	`bdi_material` INT UNSIGNED NULL,
	`bdi_installation_side` INT UNSIGNED NOT NULL,
	`bdi_type` INT UNSIGNED NOT NULL,
	`bdi_inner_diameter` INT UNSIGNED NOT NULL,
	`bdi_outer_diameter` INT UNSIGNED NOT NULL,
	`bdi_disk_width` INT UNSIGNED NOT NULL,
	`bdi_holes_number` INT UNSIGNED NOT NULL,
	`bdi_hole_width` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_brake_disc` PRIMARY KEY (`bdi_id` ASC)
)

;

CREATE TABLE `camera`
(
	`cam_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`tpr_id` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_cameras` PRIMARY KEY (`cam_id` ASC)
)

;

CREATE TABLE `candle`
(
	`can_id` INT UNSIGNED NOT NULL,
	`can_spark_gap` INT UNSIGNED NOT NULL,
	`can_caloric_num` INT UNSIGNED NOT NULL,
	`can_connecting_dimensions` INT UNSIGNED NOT NULL,
	`can_material` INT UNSIGNED NOT NULL,
	`can_num_of_side_electrodes` INT UNSIGNED NOT NULL,
	`can_is_purify` BOOL NOT NULL,
	CONSTRAINT `PK_candle` PRIMARY KEY (`can_id` ASC)
)

;

CREATE TABLE `capacity`
(
	`cap_id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
	`cap_width` INT UNSIGNED NOT NULL,
	`cap_height` INT UNSIGNED NOT NULL,
	`cap_length` INT UNSIGNED NOT NULL,
	`cap_weight` INT UNSIGNED NULL,
	CONSTRAINT `PK_capacity` PRIMARY KEY (`cap_id` ASC)
)

;

CREATE TABLE `car`
(
	`car_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`cty_id` INT UNSIGNED NOT NULL,
	`car_number` VARCHAR(30) NOT NULL,
	`use_id` INT UNSIGNED NULL,
	`war_id` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_car` PRIMARY KEY (`car_id` ASC)
)

;

CREATE TABLE `car_brand`
(
	`cba_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`cba_name` VARCHAR(150) NOT NULL,
	CONSTRAINT `PK_car_brand` PRIMARY KEY (`cba_id` ASC)
```

```
)

;

CREATE TABLE `car_type`
(
	`cty_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`cap_id` BIGINT UNSIGNED NOT NULL,
	`fac_id` INT UNSIGNED NOT NULL,
	`col_id` INT UNSIGNED NOT NULL,
	`cba_id` INT UNSIGNED NOT NULL,
	`cty_release_year` INT UNSIGNED NOT NULL,
	`cty_fuel_consumption` INT UNSIGNED NOT NULL,
	`cty_name` VARCHAR(100) NOT NULL,
	`cty_vin_num` CHAR(17) NOT NULL,
	CONSTRAINT `PK_car_type` PRIMARY KEY (`cty_id` ASC)
)

;

CREATE TABLE `color`
(
	`col_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`col_name` VARCHAR(60) NOT NULL,
	CONSTRAINT `PK_color` PRIMARY KEY (`col_id` ASC)
)

;

CREATE TABLE `comments`
(
	`com_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`pro_id` INT UNSIGNED NOT NULL,
	`use_id` INT UNSIGNED NOT NULL,
	`com_content` TEXT NOT NULL,
	CONSTRAINT `PK_comments` PRIMARY KEY (`com_id` ASC)
)

;

CREATE TABLE `delivery`
(
	`del_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`use_id` INT UNSIGNED NULL,
	`ord_id` INT UNSIGNED NOT NULL,
	`add_id` INT UNSIGNED NOT NULL,
	`del_time_from` DATETIME NOT NULL,
	`del_time_to` DATETIME NOT NULL,
	`del_time_done` DATETIME NULL,
	`del_additional_info` TEXT NULL,
	CONSTRAINT `PK_delivery` PRIMARY KEY (`del_id` ASC)
)

;

CREATE TABLE `discount`
(
	`dis_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`dis_percent` INT UNSIGNED NOT NULL,
	`dis_code` VARCHAR(16) NOT NULL,
	`dis_start_date` DATETIME NULL,
	`dis_end_date` DATETIME NULL,
	CONSTRAINT `PK_discount` PRIMARY KEY (`dis_id` ASC)
)

;

CREATE TABLE `factory`
(
```

```
	`fac_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`add_id` INT UNSIGNED NOT NULL,
	`man_id` INT UNSIGNED NOT NULL,
	`fac_name` VARCHAR(200) NOT NULL,
	CONSTRAINT `PK_manufacturer` PRIMARY KEY (`fac_id` ASC)
)

;

CREATE TABLE `filter`
(
	`fil_id` INT UNSIGNED NOT NULL,
	`fil_type` INT UNSIGNED NOT NULL,
	`fil_efficiency` INT UNSIGNED NOT NULL,
	`fil_material` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_filters` PRIMARY KEY (`fil_id` ASC)
)

;

CREATE TABLE `image_url`
(
	`iur_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`iur_url` VARCHAR(2048) NOT NULL,
	`iur_name` VARCHAR(250) NULL,
	CONSTRAINT `PK_image_url` PRIMARY KEY (`iur_id` ASC)
)

;

CREATE TABLE `m2m_mesage_user`
(
	`mes_id` INT UNSIGNED NOT NULL,
	`use_id` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_message_recipient` PRIMARY KEY (`use_id` ASC, `mes_id` ASC)
)

;

CREATE TABLE `m2m_news_image_url`
(
	`new_id` INT UNSIGNED NOT NULL,
	`iur_id` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_news_image` PRIMARY KEY (`new_id` ASC, `iur_id` ASC)
)

;

CREATE TABLE `m2m_order_product`
(
	`ord_id` INT UNSIGNED NOT NULL,
	`pro_id` INT UNSIGNED NOT NULL,
	`count` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_m2m_order_product` PRIMARY KEY (`ord_id` ASC, `pro_id` ASC)
)

;

CREATE TABLE `m2m_product_discount`
(
	`pro_id` INT UNSIGNED NOT NULL,
	`dis_id` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_m2m_product_discount` PRIMARY KEY (`pro_id` ASC, `dis_id` ASC)
)

;

CREATE TABLE `m2m_product_image_url`
(
```

```
	`pro_id` INT UNSIGNED NOT NULL,
	`iur_id` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_m2m_product_image_url` PRIMARY KEY (`pro_id` ASC, `iur_id` ASC)
)

;

CREATE TABLE `m2m_user_type_permission`
(
	`uty_id` INT UNSIGNED NOT NULL,
	`per_id` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_m2m_user_type_permission` PRIMARY KEY (`per_id` ASC, `uty_id` ASC)
)

;

CREATE TABLE `m2m_warehouse_product`
(
	`war_id` INT UNSIGNED NOT NULL,
	`pro_id` INT UNSIGNED NOT NULL,
	`count` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_m2m_warehouse_product` PRIMARY KEY (`war_id` ASC, `pro_id` ASC)
)

;

CREATE TABLE `manufacturer`
(
	`man_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`man_name` VARCHAR(200) NOT NULL,
	CONSTRAINT `PK_manufacturer` PRIMARY KEY (`man_id` ASC)
)

;

CREATE TABLE `message`
(
	`mes_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`use_id` INT UNSIGNED NOT NULL,
	`mes_content` TEXT NULL,
	`mes_time` DATETIME NOT NULL,
	CONSTRAINT `PK_messages` PRIMARY KEY (`mes_id` ASC)
)

;

CREATE TABLE `news`
(
	`new_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`use_id` INT UNSIGNED NOT NULL,
	`new_header` VARCHAR(500) NOT NULL,
	`new_content` TEXT NOT NULL,
	`new_time` DATETIME NOT NULL,
	CONSTRAINT `PK_news` PRIMARY KEY (`new_id` ASC)
)

;

CREATE TABLE `oil`
(
	`oil_id` INT UNSIGNED NOT NULL,
	`oil_type` INT UNSIGNED NOT NULL,
	`oil_capacity` INT UNSIGNED NOT NULL,
	`oil_specification` INT UNSIGNED NULL,
	CONSTRAINT `PK_oil` PRIMARY KEY (`oil_id` ASC)
)

;
```

```
CREATE TABLE `oil_seal`
(
	`ose_id` INT UNSIGNED NOT NULL,
	`ose_inner_diametr` INT UNSIGNED NOT NULL,
	`ose_outer_diametr` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_oil_seal` PRIMARY KEY (`ose_id` ASC)
)

;

CREATE TABLE `order`
(
	`ord_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`use_id` INT UNSIGNED NOT NULL,
	`ost_id` INT UNSIGNED NOT NULL,
	`ord_price` INT UNSIGNED NOT NULL,
	`ord_created_time` DATETIME NOT NULL,
	`rpo_id` INT UNSIGNED NULL,
	CONSTRAINT `PK_order` PRIMARY KEY (`ord_id` ASC)
)

;

CREATE TABLE `order_status`
(
	`ost_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`ost_name` VARCHAR(250) NOT NULL,
	CONSTRAINT `PK_order_status` PRIMARY KEY (`ost_id` ASC)
)

;

CREATE TABLE `permission`
(
	`per_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`per_name` VARCHAR(150) NOT NULL,
	CONSTRAINT `PK_permission` PRIMARY KEY (`per_id` ASC)
)

;

CREATE TABLE `product`
(
	`pro_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`pty_id` INT UNSIGNED NOT NULL,
	`cap_id` BIGINT UNSIGNED NOT NULL,
	`fac_id` INT UNSIGNED NULL,
	`col_id` INT UNSIGNED NULL,
	`cty_id` INT UNSIGNED NULL,
	`pro_name` VARCHAR(100) NOT NULL,
	`pro_description` TEXT NULL,
	`pro_price` INT NOT NULL,
	`pro_release_year` INT UNSIGNED NULL,
	CONSTRAINT `PK_product` PRIMARY KEY (`pro_id` ASC)
)

;

CREATE TABLE `product_type`
(
	`pty_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`pty_name` VARCHAR(140) NOT NULL,
	CONSTRAINT `PK_product_type` PRIMARY KEY (`pty_id` ASC)
)

;

CREATE TABLE `product_value`
(
```

```
	`pva_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`pva_value` VARCHAR(140) NOT NULL,
	`pty_id` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_product_value` PRIMARY KEY (`pva_id` ASC)
)

;

CREATE TABLE `pump`
(
	`pum_id` INT UNSIGNED NOT NULL,
	`pum_voltage` INT UNSIGNED NOT NULL,
	`pum_pressure` INT UNSIGNED NOT NULL,
	`pum_inner_diametr` INT UNSIGNED NOT NULL,
	`pum_outer_diametr` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_pump` PRIMARY KEY (`pum_id` ASC)
)

;

CREATE TABLE `reception_point`
(
	`rpo_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`add_id` INT UNSIGNED NOT NULL,
	`use_id` INT UNSIGNED NULL,
	`cap_id` BIGINT UNSIGNED NOT NULL,
	`rec_name` VARCHAR(150) NOT NULL,
	CONSTRAINT `PK_reception_point` PRIMARY KEY (`rpo_id` ASC)
)

;

CREATE TABLE `starter`
(
	`sta_id` INT UNSIGNED NOT NULL,
	`sta_type` INT UNSIGNED NOT NULL,
	`sta_voltage` INT UNSIGNED NOT NULL,
	`sta_power` INT UNSIGNED NOT NULL,
	`sta_turnovers` INT UNSIGNED NOT NULL,
	`sta_current_strength` INT UNSIGNED NOT NULL,
	`sta_tension_force` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_starter` PRIMARY KEY (`sta_id` ASC)
)

;

CREATE TABLE `steering_rack`
(
	`sra_id` INT UNSIGNED NOT NULL,
	`sra_mechanism_type` INT UNSIGNED NOT NULL,
	`sra_operation_type` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_steering_rack` PRIMARY KEY (`sra_id` ASC)
)

;

CREATE TABLE `steering_rods`
(
	`sro_id` INT UNSIGNED NOT NULL,
	`sro_rail_size` INT UNSIGNED NOT NULL,
	`sro_first_step` DOUBLE(10,2) NOT NULL,
	`sro_second_step` DOUBLE(10,2) NOT NULL,
	`sro_outer_length` INT UNSIGNED NOT NULL,
	`sro_length` INT UNSIGNED NOT NULL,
	`sro_tip` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_steering_rods` PRIMARY KEY (`sro_id` ASC)
)

;
```

```
CREATE TABLE `timing_belt`
(
	`tbe_id` INT UNSIGNED NOT NULL,
	`tbe_transport_type` INT UNSIGNED NOT NULL,
	`tbe_teeth_number` INT UNSIGNED NOT NULL,
	`tbe_profile_code` INT UNSIGNED NOT NULL,
	`tbe_teeth_type` INT UNSIGNED NOT NULL,
	`tbe_material` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_timing_belt` PRIMARY KEY (`tbe_id` ASC)
)

;

CREATE TABLE `tire`
(
	`tir_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`tpr_id` INT UNSIGNED NOT NULL,
	`tir_is_winter` BOOL NOT NULL,
	`tir_is_summer` BOOL NOT NULL,
	`tir_is_studded` BOOL NOT NULL,
	CONSTRAINT `PK_tires` PRIMARY KEY (`tir_id` ASC)
)

;

CREATE TABLE `tire_properties`
(
	`tpr_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`tpr_profile_height` INT UNSIGNED NOT NULL,
	`tpr_type` INT UNSIGNED NOT NULL,
	`tpr_diameter` INT UNSIGNED NOT NULL,
	`tpr_load_index` INT UNSIGNED NOT NULL,
	`tpr_speed_index` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_tire_properties` PRIMARY KEY (`tpr_id` ASC)
)

;

CREATE TABLE `user`
(
	`use_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`uin_id` BIGINT UNSIGNED NOT NULL,
	`use_registration_time` DATETIME NOT NULL,
	CONSTRAINT `PK_user` PRIMARY KEY (`use_id` ASC)
)

;

CREATE TABLE `user_info`
(
	`uin_id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
	`use_id` INT UNSIGNED NULL,
	`uty_id` INT UNSIGNED NOT NULL,
	`uin_email` VARCHAR(100) NOT NULL,
	`uin_password` VARCHAR(100) NOT NULL,
	`uin_nickname` VARCHAR(100) NULL,
	`uin_first_name` VARCHAR(100) NULL,
	`uin_middle_name` VARCHAR(100) NULL,
	`uin_last_name` VARCHAR(100) NULL,
	`uin_phone` VARCHAR(20) NULL,
	`uin_time_update` DATETIME NOT NULL,
	CONSTRAINT `PK_user_info` PRIMARY KEY (`uin_id` ASC)
)

;

CREATE TABLE `user_online`
(
```

```
	`uon_id` BIGINT UNSIGNED NOT NULL AUTO_INCREMENT,
	`use_id` INT UNSIGNED NOT NULL,
	`uon_time_in` DATETIME NOT NULL,
	`uon_time_out` DATETIME NOT NULL,
	CONSTRAINT `PK_user_online` PRIMARY KEY (`uon_id` ASC)
)

;

CREATE TABLE `user_type`
(
	`uty_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`uty_name` VARCHAR(150) NOT NULL,
	CONSTRAINT `PK_user_type` PRIMARY KEY (`uty_id` ASC)
)

;

CREATE TABLE `warehouse`
(
	`war_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
	`add_id` INT UNSIGNED NOT NULL,
	`cap_id` BIGINT UNSIGNED NOT NULL,
	`use_id` INT UNSIGNED NULL,
	`war_name` VARCHAR(150) NOT NULL,
	CONSTRAINT `PK_warehouse` PRIMARY KEY (`war_id` ASC)
)

;

CREATE TABLE `wheel_disc`
(
	`wdi_id` INT UNSIGNED NOT NULL,
	`wdi_diameter` INT UNSIGNED NOT NULL,
	`wdi_holes_number` INT UNSIGNED NOT NULL,
	`wdi_hole_width` INT UNSIGNED NOT NULL,
	`wdi_diameter_hole_arrangement` INT UNSIGNED NOT NULL,
	`wdi_diameter_central_hole` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_wheel_disc` PRIMARY KEY (`wdi_id` ASC)
)

;

CREATE TABLE `wiper_blades`
(
	`wbl_id` INT UNSIGNED NOT NULL,
	`wbl_size` INT UNSIGNED NOT NULL,
	`wbl_type` INT UNSIGNED NOT NULL,
	`wbl_mounting_type` INT UNSIGNED NOT NULL,
	`wbl_material` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_wiper_blades` PRIMARY KEY (`wbl_id` ASC)
)

;

CREATE TABLE `wishlist`
(
	`pro_id` INT UNSIGNED NOT NULL,
	`use_id` INT UNSIGNED NOT NULL,
	`count` INT UNSIGNED NOT NULL,
	CONSTRAINT `PK_wishlist` PRIMARY KEY (`pro_id` ASC, `use_id` ASC)
)

;

/* Create Primary Keys, Indexes, Uniques, Checks */

ALTER TABLE `accumulator`
 ADD INDEX `IXFK_accumulator_product` (`acc_id` ASC)
```

```
;

ALTER TABLE `brake_columns`
 ADD INDEX `IXFK_brake_columns_product` (`bco_id` ASC)
;

ALTER TABLE `brake_disc`
 ADD INDEX `IXFK_brake_disc_product` (`bdi_id` ASC)
;

ALTER TABLE `camera`
 ADD INDEX `IXFK_cameras_product` (`cam_id` ASC)
;

ALTER TABLE `camera`
 ADD INDEX `IXFK_cameras_tire_properties` (`tpr_id` ASC)
;

ALTER TABLE `candle`
 ADD INDEX `IXFK_candle_product` (`can_id` ASC)
;

ALTER TABLE `car_brand`
 ADD CONSTRAINT `UNX_cba_name` UNIQUE (`cba_name` ASC)
;

ALTER TABLE `car_brand`
 ADD INDEX `IX_cba_name` (`cba_name` ASC)
;

ALTER TABLE `car_type`
 ADD INDEX `IX_cty_name` (`cty_name` ASC)
;

ALTER TABLE `color`
 ADD CONSTRAINT `UNX_col_name` UNIQUE (`col_name` ASC)
;

ALTER TABLE `color`
 ADD INDEX `IX_col_name` (`col_name` ASC)
;

ALTER TABLE `filter`
 ADD INDEX `IXFK_filters_product` (`fil_id` ASC)
;

ALTER TABLE `news`
 ADD INDEX `IX_new_header` (`new_header` ASC)
;

ALTER TABLE `oil`
 ADD INDEX `IXFK_oil_product` (`oil_id` ASC)
;

ALTER TABLE `oil_seal`
 ADD INDEX `IXFK_oil_seal_product` (`ose_id` ASC)
;

DELIMITER //
CREATE TRIGGER `TRG_udpate_order_time_created_after_ins`
BEFORE INSERT
ON `order`
        FOR EACH ROW
        BEGIN
                SET NEW.`ord_created_time` = CURDATE();
        END;
//
DELIMITER ;
;
```

```
ALTER TABLE `order_status`
 ADD CONSTRAINT `UNX_ost_name` UNIQUE (`ost_name` ASC)
;

ALTER TABLE `order_status`
 ADD INDEX `IX_ost_name` (`ost_name` ASC)
;

ALTER TABLE `permission`
 ADD CONSTRAINT `UNX_per_name` UNIQUE (`per_name` ASC)
;

ALTER TABLE `product_type`
 ADD CONSTRAINT `UNX_pty_name` UNIQUE (`pty_name` ASC)
;

ALTER TABLE `product_type`
 ADD INDEX `IX_pty_name` (`pty_name` ASC)
;

ALTER TABLE `product_value`
 ADD CONSTRAINT `UNX_pva_value` UNIQUE (`pva_value` ASC)
;

ALTER TABLE `pump`
 ADD INDEX `IXFK_pump_product` (`pum_id` ASC)
;

ALTER TABLE `starter`
 ADD INDEX `IXFK_starter_product` (`sta_id` ASC)
;

ALTER TABLE `steering_rack`
 ADD INDEX `IXFK_steering_rack_product` (`sra_id` ASC)
;

ALTER TABLE `steering_rods`
 ADD INDEX `IXFK_steering_rods_product` (`sro_id` ASC)
;

ALTER TABLE `timing_belt`
 ADD INDEX `IXFK_timing_belt_product` (`tbe_id` ASC)
;

ALTER TABLE `tire`
 ADD INDEX `IXFK_tires_product` (`tir_id` ASC)
;

ALTER TABLE `tire`
 ADD INDEX `IXFK_tires_tire_properties` (`tpr_id` ASC)
;

DELIMITER //
CREATE TRIGGER `TRG_udpate_user_registration_time_before_ins`
BEFORE INSERT
ON `user`
        FOR EACH ROW
        BEGIN
                SET NEW.`use_registration_time` = CURDATE();
        END;
//
DELIMITER ;
;

ALTER TABLE `user_info`
 ADD CONSTRAINT `UNX_uin_email` UNIQUE (`uin_email` ASC)
;
```

```
ALTER TABLE `user_info`
 ADD CONSTRAINT `UNX_uin_nickname` UNIQUE (`uin_nickname` ASC)
;

ALTER TABLE `user_info`
 ADD CONSTRAINT `UNX_uin_phone` UNIQUE (`uin_phone` ASC)
;

DELIMITER //
CREATE TRIGGER `TRG_user_online_clear_before_insert`
   AFTER INSERT
   ON `user_online` FOR EACH ROW
BEGIN
   SET @count = (SELECT COUNT(*) FROM `user_online`);
        IF @count > 18446744073709551610 THEN
                CALL CLEAR_INFORMATION_ONLINE();
        END IF;
END;
//
DELIMITER ;
;

ALTER TABLE `user_type`
 ADD CONSTRAINT `UNX_uty_name` UNIQUE (`uty_name` ASC)
;

ALTER TABLE `warehouse`
 ADD INDEX `IX_war_name` (`war_name` ASC)
;

ALTER TABLE `wheel_disc`
 ADD INDEX `IXFK_wheel_disc_product` (`wdi_id` ASC)
;

ALTER TABLE `wiper_blades`
 ADD INDEX `IXFK_wiper_blades_product` (`wbl_id` ASC)
;

/* Create Foreign Key Constraints */

ALTER TABLE `accumulator`
 ADD CONSTRAINT `FK_accumulator_product`
        FOREIGN KEY (`acc_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `brake_columns`
 ADD CONSTRAINT `FK_brake_columns_product`
        FOREIGN KEY (`bco_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `brake_disc`
 ADD CONSTRAINT `FK_brake_disc_product`
        FOREIGN KEY (`bdi_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `camera`
 ADD CONSTRAINT `FK_cameras_product`
        FOREIGN KEY (`cam_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `camera`
 ADD CONSTRAINT `FK_cameras_tire_properties`
        FOREIGN KEY (`tpr_id`) REFERENCES `tire_properties` (`tpr_id`) ON DELETE No Action ON UPDATE
No Action
;

ALTER TABLE `candle`
 ADD CONSTRAINT `FK_candle_product`
        FOREIGN KEY (`can_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
```

```
;

ALTER TABLE `car`
 ADD CONSTRAINT `FK_car_car_type`
        FOREIGN KEY (`cty_id`) REFERENCES `car_type` (`cty_id`) ON DELETE Restrict ON UPDATE Cascade
;

ALTER TABLE `car`
 ADD CONSTRAINT `FK_car_user`
        FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE Set Null ON UPDATE Cascade
;

ALTER TABLE `car`
 ADD CONSTRAINT `FK_car_warehouse`
        FOREIGN KEY (`war_id`) REFERENCES `warehouse` (`war_id`) ON DELETE Restrict ON UPDATE
Restrict
;

ALTER TABLE `car_type`
 ADD CONSTRAINT `FK_car_type_capacity`
        FOREIGN KEY (`cap_id`) REFERENCES `capacity` (`cap_id`) ON DELETE Restrict ON UPDATE Cascade
;

ALTER TABLE `car_type`
 ADD CONSTRAINT `FK_car_type_car_brand`
        FOREIGN KEY (`cba_id`) REFERENCES `car_brand` (`cba_id`) ON DELETE Restrict ON UPDATE
Cascade
;

ALTER TABLE `car_type`
 ADD CONSTRAINT `FK_car_type_color`
        FOREIGN KEY (`col_id`) REFERENCES `color` (`col_id`) ON DELETE Restrict ON UPDATE Cascade
;

ALTER TABLE `car_type`
 ADD CONSTRAINT `FK_car_type_factory`
        FOREIGN KEY (`fac_id`) REFERENCES `factory` (`fac_id`) ON DELETE Restrict ON UPDATE Cascade
;

ALTER TABLE `comments`
 ADD CONSTRAINT `FK_comments_product`
        FOREIGN KEY (`pro_id`) REFERENCES `product` (`pro_id`) ON DELETE No Action ON UPDATE No
Action
;

ALTER TABLE `comments`
 ADD CONSTRAINT `FK_comments_user`
        FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE No Action ON UPDATE Cascade
;

ALTER TABLE `delivery`
 ADD CONSTRAINT `FK_delivery_address`
        FOREIGN KEY (`add_id`) REFERENCES `address` (`add_id`) ON DELETE Restrict ON UPDATE Cascade
;

ALTER TABLE `delivery`
 ADD CONSTRAINT `FK_delivery_order`
        FOREIGN KEY (`ord_id`) REFERENCES `order` (`ord_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `delivery`
 ADD CONSTRAINT `FK_delivery_user`
        FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE Restrict ON UPDATE Restrict
;

ALTER TABLE `factory`
 ADD CONSTRAINT `FK_factory_manufacturer`
        FOREIGN KEY (`man_id`) REFERENCES `manufacturer` (`man_id`) ON DELETE Restrict ON UPDATE
Cascade
```

```
;

ALTER TABLE `factory`
 ADD CONSTRAINT `FK_manufacturer_address`
        FOREIGN KEY (`add_id`) REFERENCES `address` (`add_id`) ON DELETE Restrict ON UPDATE Cascade
;

ALTER TABLE `filter`
 ADD CONSTRAINT `FK_filters_product`
        FOREIGN KEY (`fil_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `m2m_mesage_user`
 ADD CONSTRAINT `FK_message_recipient_message`
        FOREIGN KEY (`mes_id`) REFERENCES `message` (`mes_id`) ON DELETE Cascade ON UPDATE
Cascade
;

ALTER TABLE `m2m_mesage_user`
 ADD CONSTRAINT `FK_message_recipient_user`
        FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `m2m_news_image_url`
 ADD CONSTRAINT `FK_m2m_news_image_url_image_url`
        FOREIGN KEY (`iur_id`) REFERENCES `image_url` (`iur_id`) ON DELETE Cascade ON UPDATE
Cascade
;

ALTER TABLE `m2m_news_image_url`
 ADD CONSTRAINT `FK_m2m_news_image_url_news`
        FOREIGN KEY (`new_id`) REFERENCES `news` (`new_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `m2m_order_product`
 ADD CONSTRAINT `FK_m2m_order_product_order`
        FOREIGN KEY (`ord_id`) REFERENCES `order` (`ord_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `m2m_order_product`
 ADD CONSTRAINT `FK_m2m_order_product_product`
        FOREIGN KEY (`pro_id`) REFERENCES `product` (`pro_id`) ON DELETE Restrict ON UPDATE Cascade
;

ALTER TABLE `m2m_product_discount`
 ADD CONSTRAINT `FK_m2m_product_discount_discount`
        FOREIGN KEY (`dis_id`) REFERENCES `discount` (`dis_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `m2m_product_discount`
 ADD CONSTRAINT `FK_m2m_product_discount_product`
        FOREIGN KEY (`pro_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `m2m_product_image_url`
 ADD CONSTRAINT `FK_m2m_product_image_url_image_url`
        FOREIGN KEY (`iur_id`) REFERENCES `image_url` (`iur_id`) ON DELETE Cascade ON UPDATE
Cascade
;

ALTER TABLE `m2m_product_image_url`
 ADD CONSTRAINT `FK_m2m_product_image_url_product`
        FOREIGN KEY (`pro_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `m2m_user_type_permission`
 ADD CONSTRAINT `FK_m2m_user_type_permission_permission`
        FOREIGN KEY (`per_id`) REFERENCES `permission` (`per_id`) ON DELETE Cascade ON UPDATE
Cascade
```

```
;

ALTER TABLE `m2m_user_type_permission`
 ADD CONSTRAINT `FK_m2m_user_type_permission_user_type`
        FOREIGN KEY (`uty_id`) REFERENCES `user_type` (`uty_id`) ON DELETE Cascade ON UPDATE
Cascade
;

ALTER TABLE `m2m_warehouse_product`
 ADD CONSTRAINT `FK_m2m_warehouse_product_product`
        FOREIGN KEY (`pro_id`) REFERENCES `product` (`pro_id`) ON DELETE Restrict ON UPDATE Cascade
;

ALTER TABLE `m2m_warehouse_product`
 ADD CONSTRAINT `FK_m2m_warehouse_product_warehouse`
        FOREIGN KEY (`war_id`) REFERENCES `warehouse` (`war_id`) ON DELETE Restrict ON UPDATE
Cascade
;

ALTER TABLE `message`
 ADD CONSTRAINT `FK_message_user`
        FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE No Action ON UPDATE No Action
;

ALTER TABLE `news`
 ADD CONSTRAINT `FK_news_user`
        FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE No Action ON UPDATE Cascade
;

ALTER TABLE `oil`
 ADD CONSTRAINT `FK_oil_product`
        FOREIGN KEY (`oil_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `oil_seal`
 ADD CONSTRAINT `FK_oil_seal_product`
        FOREIGN KEY (`ose_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `order`
 ADD CONSTRAINT `FK_order_order_status`
        FOREIGN KEY (`ost_id`) REFERENCES `order_status` (`ost_id`) ON DELETE Restrict ON UPDATE
Restrict
;

ALTER TABLE `order`
 ADD CONSTRAINT `FK_order_reception_point`
        FOREIGN KEY (`rpo_id`) REFERENCES `reception_point` (`rpo_id`) ON DELETE Restrict ON UPDATE
Restrict
;

ALTER TABLE `order`
 ADD CONSTRAINT `FK_order_user`
        FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `product`
 ADD CONSTRAINT `FK_product_capacity`
        FOREIGN KEY (`cap_id`) REFERENCES `capacity` (`cap_id`) ON DELETE Restrict ON UPDATE Cascade
;

ALTER TABLE `product`
 ADD CONSTRAINT `FK_product_car_type`
        FOREIGN KEY (`cty_id`) REFERENCES `car_type` (`cty_id`) ON DELETE Restrict ON UPDATE Restrict
;

ALTER TABLE `product`
 ADD CONSTRAINT `FK_product_color`
        FOREIGN KEY (`col_id`) REFERENCES `color` (`col_id`) ON DELETE Restrict ON UPDATE Cascade
```

```
;

ALTER TABLE `product`
 ADD CONSTRAINT `FK_product_factory`
        FOREIGN KEY (`fac_id`) REFERENCES `factory` (`fac_id`) ON DELETE No Action ON UPDATE No
Action
;

ALTER TABLE `product`
 ADD CONSTRAINT `FK_product_product_type`
        FOREIGN KEY (`pty_id`) REFERENCES `product_type` (`pty_id`) ON DELETE Restrict ON UPDATE
Restrict
;

ALTER TABLE `product_value`
 ADD CONSTRAINT `FK_product_value_product_type`
        FOREIGN KEY (`pty_id`) REFERENCES `product_type` (`pty_id`) ON DELETE Restrict ON UPDATE
Restrict
;

ALTER TABLE `pump`
 ADD CONSTRAINT `FK_pump_product`
        FOREIGN KEY (`pum_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE
Cascade
;

ALTER TABLE `reception_point`
 ADD CONSTRAINT `FK_reception_point_address`
        FOREIGN KEY (`add_id`) REFERENCES `address` (`add_id`) ON DELETE Restrict ON UPDATE Cascade
;

ALTER TABLE `reception_point`
 ADD CONSTRAINT `FK_reception_point_capacity`
        FOREIGN KEY (`cap_id`) REFERENCES `capacity` (`cap_id`) ON DELETE Restrict ON UPDATE Restrict
;

ALTER TABLE `reception_point`
 ADD CONSTRAINT `FK_reception_point_user`
        FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE Restrict ON UPDATE Restrict
;

ALTER TABLE `starter`
 ADD CONSTRAINT `FK_starter_product`
        FOREIGN KEY (`sta_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `steering_rack`
 ADD CONSTRAINT `FK_steering_rack_product`
        FOREIGN KEY (`sra_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `steering_rods`
 ADD CONSTRAINT `FK_steering_rods_product`
        FOREIGN KEY (`sro_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `timing_belt`
 ADD CONSTRAINT `FK_timing_belt_product`
        FOREIGN KEY (`tbe_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `tire`
 ADD CONSTRAINT `FK_tires_product`
        FOREIGN KEY (`tir_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `tire`
 ADD CONSTRAINT `FK_tires_tire_properties`
```

```
	FOREIGN KEY (`tpr_id`) REFERENCES `tire_properties` (`tpr_id`) ON DELETE No Action ON UPDATE
No Action
;

ALTER TABLE `user`
 ADD CONSTRAINT `FK_user_user_info`
	FOREIGN KEY (`uin_id`) REFERENCES `user_info` (`uin_id`) ON DELETE Restrict ON UPDATE
Cascade
;

ALTER TABLE `user_info`
 ADD CONSTRAINT `FK_user_info_user`
	FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `user_info`
 ADD CONSTRAINT `FK_user_info_user_type`
	FOREIGN KEY (`uty_id`) REFERENCES `user_type` (`uty_id`) ON DELETE Restrict ON UPDATE No
Action
;

ALTER TABLE `user_online`
 ADD CONSTRAINT `FK_user_online_user`
	FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `warehouse`
 ADD CONSTRAINT `FK_warehouse_address`
	FOREIGN KEY (`add_id`) REFERENCES `address` (`add_id`) ON DELETE No Action ON UPDATE No
Action
;

ALTER TABLE `warehouse`
 ADD CONSTRAINT `FK_warehouse_capacity`
	FOREIGN KEY (`cap_id`) REFERENCES `capacity` (`cap_id`) ON DELETE No Action ON UPDATE No
Action
;

ALTER TABLE `warehouse`
 ADD CONSTRAINT `FK_warehouse_user`
	FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE Restrict ON UPDATE Restrict
;

ALTER TABLE `wheel_disc`
 ADD CONSTRAINT `FK_wheel_disc_product`
	FOREIGN KEY (`wdi_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `wiper_blades`
 ADD CONSTRAINT `FK_wiper_blades_product`
	FOREIGN KEY (`wbl_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `wishlist`
 ADD CONSTRAINT `FK_wishlist_product`
	FOREIGN KEY (`pro_id`) REFERENCES `product` (`pro_id`) ON DELETE Cascade ON UPDATE Cascade
;

ALTER TABLE `wishlist`
 ADD CONSTRAINT `FK_wishlist_user`
	FOREIGN KEY (`use_id`) REFERENCES `user` (`use_id`) ON DELETE Cascade ON UPDATE Cascade
;

SET FOREIGN_KEY_CHECKS=1
;
/* Create Views */

CREATE OR REPLACE VIEW GET_ACCUMULATORS AS
  SELECT `product`.*, `col_name`,
```

```
                    `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                    GET_PRODUCT_VALUE_BY_ID(`acc_type`) AS `acc_type`,
        `acc_capacity`, `acc_voltage`, `acc_efficiency`,
        `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
        `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `accumulator` ON `acc_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_BRAKE_COLUMNS AS
    SELECT `product`.*, `col_name`,
                    `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                        GET_PRODUCT_VALUE_BY_ID(`bco_frame_material`) AS `bco_frame_material`,
                        GET_PRODUCT_VALUE_BY_ID(`bco_internal_material`) AS `bco_internal_material`,
        `bco_column_width`, `bco_column_length`, `bco_friction_coefficient`, `bco_max_load`,
        `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`,
        `add_postcode`, `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `brake_columns` ON `bco_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_BRAKE_DISCS AS
    SELECT `product`.*, `col_name`,
                    `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                    `bdi_is_ventilated`,
                    GET_PRODUCT_VALUE_BY_ID(`bdi_perforation_type`) AS `bdi_perforation_type`,
                    GET_PRODUCT_VALUE_BY_ID(`bdi_material`) AS `bdi_material`,
                    GET_PRODUCT_VALUE_BY_ID(`bdi_installation_side`) AS `bdi_installation_side`,
                    GET_PRODUCT_VALUE_BY_ID(`bdi_type`) AS `bdi_type`,
        `bdi_inner_diameter`, `bdi_outer_diameter`, `bdi_disk_width`,
        `bdi_holes_number`, `bdi_hole_width`, `fac_name`, `man_name`,
        `add_country`, `add_city`, `add_region`, `add_street`, `add_building`,
        `add_home`, `add_apartment`, `add_postcode`, `cap_width`, `cap_height`,
        `cap_length`, `cap_weight`
    FROM `product`
    JOIN `brake_disc` ON `bdi_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_CAMERAS AS
    SELECT `product`.*, `camera`.*, `col_name`,
                    `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                    `fac_name`, `man_name`, `tpr_profile_height`,
        GET_PRODUCT_VALUE_BY_ID(`tpr_type`) AS `tpr_type`,
        `tpr_diameter`, `tpr_load_index`, `tpr_speed_index`,
        `add_country`, `add_city`, `add_region`, `add_street`,
        `add_building`, `add_home`, `add_apartment`, `add_postcode`,
```

```
        `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `camera` ON `cam_id` = `pro_id`
    JOIN `tire_properties` USING(`tpr_id`)
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_CANDLES AS
    SELECT `product`.*, `col_name`, `can_spark_gap`,
                        `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                        `can_caloric_num`, `can_connecting_dimensions`,
                        GET_PRODUCT_VALUE_BY_ID(`can_material`) AS `can_material`,
        `can_num_of_side_electrodes`, `can_is_purify`,
        `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
        `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `candle` ON `can_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_FILTERS AS
    SELECT `product`.*, `col_name`,
                        `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                        GET_PRODUCT_VALUE_BY_ID(`fil_type`) AS `fil_type`,
        `fil_efficiency`,
                        GET_PRODUCT_VALUE_BY_ID(`fil_material`) AS `fil_material`,
        `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
        `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `filter` ON `fil_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_OIL_SEALS AS
    SELECT `product`.*, `col_name`, `oil_seal`.*,
                        `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
        `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
        `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `oil_seal` ON `ose_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);
;
```

```
CREATE OR REPLACE VIEW GET_OILS AS
   SELECT `product`.*, `col_name`,
                        `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                        GET_PRODUCT_VALUE_BY_ID(`oil_type`) AS `oil_type`,
        `oil_capacity`,
                        GET_PRODUCT_VALUE_BY_ID(`oil_specification`) AS `oil_specification`,
        `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
        `cap_width`, `cap_height`, `cap_length`, `cap_weight`
   FROM `product`
   JOIN `oil` ON `oil_id` = `pro_id`
   JOIN `color` USING(`col_id`)
   JOIN `factory` USING(`fac_id`)
   JOIN `manufacturer` USING(`man_id`)
   JOIN `address` USING(`add_id`)
   JOIN `capacity` USING(`cap_id`)
   JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_ORDERS AS
            SELECT `ord_price`, `ord_created_time`, `product`.*,
                        `count`, `address`.*, `del_time_from`, `del_time_to`,
        `del_time_done`, `del_additional_info`
   FROM `order`
            JOIN `m2m_order_product` USING(`ord_id`)
            JOIN `product` USING(`pro_id`)
            JOIN `delivery` USING(`ord_id`)
            JOIN `address` USING(`add_id`);
;


CREATE OR REPLACE VIEW GET_PUMPS AS
   SELECT `product`.*, `col_name`, `pump`.*,
                        `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
        `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
        `cap_width`, `cap_height`, `cap_length`, `cap_weight`
   FROM `product`
   JOIN `pump` ON `pum_id` = `pro_id`
   JOIN `color` USING(`col_id`)
   JOIN `factory` USING(`fac_id`)
   JOIN `manufacturer` USING(`man_id`)
   JOIN `address` USING(`add_id`)
   JOIN `capacity` USING(`cap_id`)
   JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_STARTERS AS
   SELECT `product`.*, `col_name`,
                        `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                        GET_PRODUCT_VALUE_BY_ID(`sta_type`) AS `sta_type`,
        `sta_voltage`, `sta_power`, `sta_turnovers`,
        `sta_current_strength`, `sta_tension_force`,
        `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
        `cap_width`, `cap_height`, `cap_length`, `cap_weight`
   FROM `product`
   JOIN `starter` ON `sta_id` = `pro_id`
   JOIN `color` USING(`col_id`)
   JOIN `factory` USING(`fac_id`)
   JOIN `manufacturer` USING(`man_id`)
   JOIN `address` USING(`add_id`)
   JOIN `capacity` USING(`cap_id`)
   JOIN `car_type` USING(`cty_id`);
;
```

```
CREATE OR REPLACE VIEW GET_STEERING_RACKS AS
   SELECT `product`.*, `col_name`,
                        `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                        GET_PRODUCT_VALUE_BY_ID(`sra_mechanism_type`) AS `sra_mechanism_type`,
                        GET_PRODUCT_VALUE_BY_ID(`sra_operation_type`) AS `sra_operation_type`,
        `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
        `cap_width`, `cap_height`, `cap_length`, `cap_weight`
   FROM `product`
   JOIN `steering_rack` ON `sra_id` = `pro_id`
   JOIN `color` USING(`col_id`)
   JOIN `factory` USING(`fac_id`)
   JOIN `manufacturer` USING(`man_id`)
   JOIN `address` USING(`add_id`)
   JOIN `capacity` USING(`cap_id`)
   JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_STEERING_RODS AS
   SELECT `product`.*, `col_name`, `steering_rods`.*,
                        `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
        `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
        `cap_width`, `cap_height`, `cap_length`, `cap_weight`
   FROM `product`
   JOIN `steering_rods` ON `sro_id` = `pro_id`
   JOIN `color` USING(`col_id`)
   JOIN `factory` USING(`fac_id`)
   JOIN `manufacturer` USING(`man_id`)
   JOIN `address` USING(`add_id`)
   JOIN `capacity` USING(`cap_id`)
   JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_TIMING_BELTS AS
   SELECT `product`.*, `col_name`,
                                `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                                GET_PRODUCT_VALUE_BY_ID(`tbe_transport_type`) AS `tbe_transport_type`,
        `tbe_teeth_number`,
                                GET_PRODUCT_VALUE_BY_ID(`tbe_profile_code`) AS `tbe_profile_code`,
                                GET_PRODUCT_VALUE_BY_ID(`tbe_teeth_type`) AS `tbe_teeth_type`,
        `tbe_material`, `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
        `add_street`, `add_building`, `add_home`, `add_apartment`,
        `add_postcode`, `cap_width`, `cap_height`, `cap_length`, `cap_weight`
   FROM `product`
   JOIN `timing_belt` ON `tbe_id` = `pro_id`
   JOIN `color` USING(`col_id`)
   JOIN `factory` USING(`fac_id`)
   JOIN `manufacturer` USING(`man_id`)
   JOIN `address` USING(`add_id`)
   JOIN `capacity` USING(`cap_id`)
   JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_TIRES AS
   SELECT `product`.*, `tire`.*, `col_name`,
                        `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                        `fac_name`, `man_name`, `tpr_profile_height`,
       GET_PRODUCT_VALUE_BY_ID(`tpr_type`) AS `tpr_type`,
       `tpr_diameter`,
                        GET_PRODUCT_VALUE_BY_ID(`tpr_load_index`) AS `tpr_load_index`,
                        GET_PRODUCT_VALUE_BY_ID(`tpr_speed_index`) AS `tpr_speed_index`,
       `add_country`, `add_city`, `add_region`, `add_street`,
       `add_building`, `add_home`, `add_apartment`, `add_postcode`,
       `cap_width`, `cap_height`, `cap_length`, `cap_weight`
```

```
  FROM `product`
  JOIN `tire` ON `tir_id` = `pro_id`
  JOIN `tire_properties` USING(`tpr_id`)
  JOIN `color` USING(`col_id`)
  JOIN `factory` USING(`fac_id`)
  JOIN `manufacturer` USING(`man_id`)
  JOIN `address` USING(`add_id`)
  JOIN `capacity` USING(`cap_id`)
  JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_TOP_5_DELIVERY_USER_BY_NUM_OF_ORDERS AS
        SELECT `use_id`, COUNT(`use_id`) AS `count`
        FROM `order`
        GROUP BY `use_id`
        ORDER BY COUNT(`count`) DESC LIMIT 5;
;


CREATE OR REPLACE VIEW GET_TOP_5_ORDERS_BY_PRICE AS
        SELECT * FROM `order`
        ORDER BY `ord_price` DESC LIMIT 5;
;


CREATE OR REPLACE VIEW GET_TOP_5_PRODUCT_BY_ORDER AS
        SELECT `pro_id`, SUM(`count`) AS `count`
        FROM `m2m_order_product`
        GROUP BY `pro_id`
        ORDER BY COUNT(`count`) DESC LIMIT 5;
;


CREATE OR REPLACE VIEW GET_USERS_INFO AS
        SELECT `use_registration_time`, `user_info`.*
  FROM `user`
        JOIN `user_info` USING(`use_id`);
;


CREATE OR REPLACE VIEW GET_WHEEL_DISCS AS
  SELECT `product`.*, `wheel_disc`.*, `col_name`,
                `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
      `add_street`, `add_building`, `add_home`, `add_apartment`,
      `add_postcode`, `cap_width`, `cap_height`, `cap_length`, `cap_weight`
  FROM `product`
  JOIN `wheel_disc` ON `wdi_id` = `pro_id`
  JOIN `color` USING(`col_id`)
  JOIN `factory` USING(`fac_id`)
  JOIN `manufacturer` USING(`man_id`)
  JOIN `address` USING(`add_id`)
  JOIN `capacity` USING(`cap_id`)
  JOIN `car_type` USING(`cty_id`);
;


CREATE OR REPLACE VIEW GET_WIPER_BLADES AS
  SELECT `product`.*, `col_name`, `wbl_size`,
                `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
                GET_PRODUCT_VALUE_BY_ID(`wbl_type`) AS `wbl_type`,
      `wbl_mounting_type`,
                GET_PRODUCT_VALUE_BY_ID(`wbl_material`) AS `wbl_material`,
      `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
      `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
      `cap_width`, `cap_height`, `cap_length`, `cap_weight`
  FROM `product`
  JOIN `wiper_blades` ON `wbl_id` = `pro_id`
```

```
   JOIN `color` USING(`col_id`)
   JOIN `factory` USING(`fac_id`)
   JOIN `manufacturer` USING(`man_id`)
   JOIN `address` USING(`add_id`)
   JOIN `capacity` USING(`cap_id`)
   JOIN `car_type` USING(`cty_id`);
;


/* Create Stored Procedures */

DELIMITER //
CREATE PROCEDURE CLEAR_INFORMATION_ONLINE()
BEGIN
            DELETE FROM `user_online`
            WHERE `uon_out` IS NOT NULL;
END;
//
DELIMITER ;

;

DELIMITER //
CREATE PROCEDURE UPDATE_USER_INFO(
   `user_id` INTEGER UNSIGNED,
            `user_type` INTEGER UNSIGNED,
   `email` VARCHAR(100),
   `password` VARCHAR(100),
   `nickname` VARCHAR(100),
   `first_name` VARCHAR(100),
   `middle_name` VARCHAR(100),
   `last_name` VARCHAR(100),
   `phone` VARCHAR(20)
)
BEGIN
            INSERT INTO `user_info`
   VALUES(
      NULL,
      `user_id`,
      `user_type`,
      `email`,
      `password`,
      `nickname`,
      `first_name`,
      `middle_name`,
      `last_name`,
      `phone`,
      CURDATE()
   );
   UPDATE `user_info` SET `use_id` = `user_id` WHERE `uin_id` = LAST_INSERT_ID();
END;
//
DELIMITER ;

;
```

5 ОПИСАНИЕ БИЗНЕСЛОГИКИ

5.1 Функции базы данных

В базе данных имеются 5 функции:

– получение уникального идентификатора значения продукта;

– получение значения продукта по уникальному идентификатору;

– получение информации о проведенном времени пользователем на сайте;

– создание заказа;

– создание пользователя.

Функция получения уникального идентификатора значения продукта получает на вход значение продукта, а также его тип. Данная функция необходима для обеспечения безопасности и целостности при вводе различных значений (рисунок 5.1).

```
CREATE FUNCTION GET_ID_BY_PRODUCT_VALUE(
    `val` VARCHAR(140),
    `prod_type` INTEGER UNSIGNED
)
RETURNS INTEGER UNSIGNED DETERMINISTIC
    RETURN (
        SELECT `pva_id`
        FROM `product_value` `pv`
        WHERE `val` = `pv`.`pva_value` AND `prod_type` = `pv`.`pty_id`
    );

```

Рисунок 5.1 – Код функции получения уникального идентификатора значения продукта

Функция получения значения продукта по уникальному идентификатору (рисунок 5.2).

```
CREATE FUNCTION GET_PRODUCT_VALUE_BY_ID(`product_val_id` INTEGER UNSIGNED)
RETURNS VARCHAR(140) DETERMINISTIC
    RETURN (
        SELECT `pva_value`
        FROM `product_value`
        WHERE `product_val_id` = `pva_id`
    );
```

Рисунок 5.2 – Код функции получения значения продукта по уникальному идентификатору

Функция получения информации о проведенном времени пользователем на сайте. Принимает на вход идентификатор пользователя (рисунок 5.3).

```sql
CREATE FUNCTION GET_USER_TOTAL_TIME_ON_SITE(`user_id` INTEGER UNSIGNED)
RETURNS INTEGER UNSIGNED DETERMINISTIC
    RETURN (
        SELECT
            IFNULL(
                SUM(
                    TIME_TO_SEC(`uon_time_out`) -
                    TIME_TO_SEC(`uon_time_in`)
                ), 0
            )
        FROM `user_online`
        WHERE `use_id` = `user_id`
    );

```

Рисунок 5.3 – Код функции получения информации о проведенном времени пользователем на сайте

Функция создания заказа принимает на вход: идентификатор покупателя, идентификатор доставщика, идентификатор статуса, цену, идентификатор адреса, с какого времени осуществлять доставку, по какое время необходимо осуществить доставку и дополнительную информацию. После выполнения функция возвращает идентификатор созданного заказа (рисунок 5.4).

```sql
CREATE FUNCTION CREATE_ORDER(
    `user` INTEGER UNSIGNED,
    `delivery_user` INTEGER UNSIGNED,
    `status` INTEGER UNSIGNED,
    `price` INTEGER UNSIGNED,
    `address` INTEGER UNSIGNED,
    `time_from` DATETIME,
    `time_to` DATETIME,
    `additional_info` TEXT
)
RETURNS INTEGER UNSIGNED DETERMINISTIC
BEGIN
    INSERT INTO `order`
    VALUES(NULL, `user`, `status`, `price`, CURDATE());

    INSERT INTO `delivery`
    VALUES(NULL, `user`, LAST_INSERT_ID(), `address`, `time_from`, `time_to`, NULL, `additional_info`);

    RETURN LAST_INSERT_ID();
END;
```

Рисунок 5.4 – Код функции создания заказа

Функция создания пользователя принимает на вход: идентификатор типа пользователя, электронную почту, пароль, прозвище, имя, отчество, фамилию, телефон. После выполнения функция возвращает идентификатор созданного пользоватедя (рисунок 5.5).

```sql
CREATE FUNCTION CREATE_USER(
    `user_type` INTEGER UNSIGNED,
    `email` VARCHAR(100),
    `password` VARCHAR(100),
    `nickname` VARCHAR(100),
    `first_name` VARCHAR(100),
    `middle_name` VARCHAR(100),
    `last_name` VARCHAR(100),
    `phone` VARCHAR(20)
)
RETURNS INTEGER UNSIGNED DETERMINISTIC
BEGIN
    INSERT INTO `user_info`
    VALUES(
        NULL,
        NULL,
        `user_type`,
        `email`,
        `password`,
        `nickname`,
        `first_name`,
        `middle_name`,
        `last_name`,
        `phone`,
        CURDATE()
    );

    SET @user_info_id = LAST_INSERT_ID();
    INSERT INTO `user` VALUES(NULL, user_info_id, CURDATE());
    UPDATE `user_info` SET `use_id` = LAST_INSERT_ID() WHERE `uin_id` = @user_info_id;

    RETURN LAST_INSERT_ID();
END;
```

Рисунок 5.5 – Код функции создания пользователя

5.2 Процедуры базы данных

В базе данных имеются 2 процедуры:

– обновление информации пользователя;

– очистка статистики посещения пользователей.

Процедура обновления информации пользователя. В данную процедуру передаются следующие параметры: идентификатор пользователя, идентификатор типа, электронная почта, пароль, прозвище, имя, отчество, фамилия, телефон (рисунок 5.6).

```
CREATE PROCEDURE UPDATE_USER_INFO(
    `user_id` INTEGER UNSIGNED,
    `user_type` INTEGER UNSIGNED,
    `email` VARCHAR(100),
    `password` VARCHAR(100),
    `nickname` VARCHAR(100),
    `first_name` VARCHAR(100),
    `middle_name` VARCHAR(100),
    `last_name` VARCHAR(100),
    `phone` VARCHAR(20)
)
BEGIN
    INSERT INTO `user_info`
    VALUES(
        NULL,
        `user_id`,
        `user_type`,
        `email`,
        `password`,
        `nickname`,
        `first_name`,
        `middle_name`,
        `last_name`,
        `phone`,
        CURDATE()
    );
    UPDATE `user_info` SET `use_id` = `user_id` WHERE `uin_id` = LAST_INSERT_ID();
END;


```

Рисунок 5.6 – Код процедуры обновления информации пользователя

Процедура очистки статистики посещения пользователей (рисунок 5.7).

```
CREATE PROCEDURE CLEAR_INFORMATION_ONLINE()
BEGIN
    DELETE FROM `user_online`
    WHERE `uon_out` IS NOT NULL;
END;

```

Рисунок 5.7 – Код процедуры очистки статистики посещения пользователей

5.3 Представления базы данных

В базе данных имеется 21 представление:
– получить все заказы;
– получить информацию о пользователях;
– получить 5 самых дорогих заказов;
– получить 5 лучших доставщиков по количеству заказов;
– получить 5 лучших продукта по количеству заказов;

– получить все масла;
– получить все рулевые рейки;
– получить все рулевые тяги;
– получить все дворники;
– получить все свечи;
– получить все сальники;
– получить все фильтра;
– получить все стартера;
– получить все помпы;
– получить все аккумуляторы;
– получить все тормозные диски;
– получить все тормозные колонки;
– получить все зубчатые ремни;
– получить все колесные диски;
– получить все камеры;
– получить все шины.

Представление всех заказов, которое используется для формирования статистки на сервере (рисунок 5.8).

```
CREATE OR REPLACE VIEW GET_ORDERS AS
    SELECT `ord_price`, `ord_created_time`, `product`.*,
           `count`, `address`.*, `del_time_from`, `del_time_to`,
           `del_time_done`, `del_additional_info`
    FROM `order`
    JOIN `m2m_order_product` USING(`ord_id`)
    JOIN `product` USING(`pro_id`)
    JOIN `delivery` USING(`ord_id`)
    JOIN `address` USING(`add_id`);

```

Рисунок 5.8 – Код представления всех заказов

Представление всей информаци пользователей. Используется для поиска пользователей (рисунок 5.9).

```
CREATE OR REPLACE VIEW GET_USERS_INFO AS
    SELECT `use_registration_time`, `user_info`.*
    FROM `user`
    JOIN `user_info` USING(`use_id`);

```

Рисунок 5.9 – Код представления всей информации пользователей

Представление 5 самых дорогих заказа. Данное представление используется для отображения статистики заказов на странице директора (рисунок 5.10).

```sql
CREATE OR REPLACE VIEW GET_TOP_5_ORDERS_BY_PRICE AS
    SELECT * FROM `order`
    ORDER BY `ord_price` DESC LIMIT 5;

```

Рисунок 5.10 – Код представления 5 самых дорогих заказа

Представление 5 лучших доставщика по количеству заказов. Это представление помогает определить лучших доставщиков (рисунок 5.11).

```sql
CREATE OR REPLACE VIEW GET_TOP_5_DELIVERY_USER_BY_NUM_OF_ORDERS AS
    SELECT `use_id`, COUNT(`use_id`) AS `count`
    FROM `order`
    GROUP BY `use_id`
    ORDER BY COUNT(`count`) DESC LIMIT 5;
```

Рисунок 5.11 – Код представления 5 лучших доставщика по количеству заказов

Представление 5 лучших продукта по количеству заказов. С помощью данного представления эти продукты отображаются на главной странице магазина (рисунок 5.12).

```sql
CREATE OR REPLACE VIEW GET_TOP_5_PRODUCT_BY_ORDER AS
    SELECT `pro_id`, SUM(`count`) AS `count`
    FROM `m2m_order_product`
    GROUP BY `pro_id`
    ORDER BY COUNT(`count`) DESC LIMIT 5;

```

Рисунок 5.12 – Код представления 5 лучших продукта по количеству заказов

Представлние служащее для выбора всех масел (рисунок 5.13).

```sql
CREATE OR REPLACE VIEW GET_OILS AS
    SELECT `product`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           GET_PRODUCT_VALUE_BY_ID(`oil_type`) AS `oil_type`,
           `oil_capacity`,
           GET_PRODUCT_VALUE_BY_ID(`oil_specification`) AS `oil_specification`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `oil` ON `oil_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.13 – Код представления масел

Представлние служащее для выбора всех рулевых реек (рисунок 5.14).

```sql
CREATE OR REPLACE VIEW GET_STEERING_RACKS AS
    SELECT `product`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           GET_PRODUCT_VALUE_BY_ID(`sra_mechanism_type`) AS `sra_mechanism_type`,
           GET_PRODUCT_VALUE_BY_ID(`sra_operation_type`) AS `sra_operation_type`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `steering_rack` ON `sra_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.14 – Код представления рулевых реек

Представлние служащее для выбора всех рулевых тяг (рисунок 5.15).

```sql
CREATE OR REPLACE VIEW GET_STEERING_RODS AS
    SELECT `product`.*, `col_name`, `steering_rods`.*,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `steering_rods` ON `sro_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.15 – Код представления рулевых тяг

Представлние служащее для выбора всех дворников (рисунок 5.16).

```sql
CREATE OR REPLACE VIEW GET_WIPER_BLADES AS
    SELECT `product`.*, `col_name`, `wbl_size`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           GET_PRODUCT_VALUE_BY_ID(`wbl_type`) AS `wbl_type`,
           `wbl_mounting_type`,
           GET_PRODUCT_VALUE_BY_ID(`wbl_material`) AS `wbl_material`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `wiper_blades` ON `wbl_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.16 – Код представления дворников

Представлние служащее для выбора всех свечей (рисунок 5.17).

```sql
CREATE OR REPLACE VIEW GET_CANDLES AS
    SELECT `product`.*, `col_name`, `can_spark_gap`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           `can_caloric_num`, `can_connecting_dimensions`,
           GET_PRODUCT_VALUE_BY_ID(`can_material`) AS `can_material`,
           `can_num_of_side_electrodes`, `can_is_purify`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `candle` ON `can_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.17 – Код представления свечей

Представлние служащее для выбора всех сальников (рисунок 5.18).

```sql
CREATE OR REPLACE VIEW GET_OIL_SEALS AS
    SELECT `product`.*, `col_name`, `oil_seal`.*,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `oil_seal` ON `ose_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.18 – Код представления сальников

Представлние служащее для выбора всех фильтров (рисунок 5.19).

```sql
CREATE OR REPLACE VIEW GET_FILTERS AS
    SELECT `product`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           GET_PRODUCT_VALUE_BY_ID(`fil_type`) AS `fil_type`,
           `fil_efficiency`,
           GET_PRODUCT_VALUE_BY_ID(`fil_material`) AS `fil_material`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `filter` ON `fil_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.19 – Код представления фильтров

Представлние служащее для выбора всех стартеров (рисунок 5.20).

```sql
CREATE OR REPLACE VIEW GET_STARTERS AS
    SELECT `product`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           GET_PRODUCT_VALUE_BY_ID(`sta_type`) AS `sta_type`,
           `sta_voltage`, `sta_power`, `sta_turnovers`,
           `sta_current_strength`, `sta_tension_force`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `starter` ON `sta_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.20 – Код представления стартеров

Представлние служащее для выбора всех помп (рисунок 5.21).

```sql
CREATE OR REPLACE VIEW GET_PUMPS AS
    SELECT `product`.*, `col_name`, `pump`.*,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `pump` ON `pum_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.21 – Код представления помп

Представлние служащее для выбора всех аккумуляторов (рисунок 5.22).

```sql
CREATE OR REPLACE VIEW GET_ACCUMULATORS AS
    SELECT `product`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           GET_PRODUCT_VALUE_BY_ID(`acc_type`) AS `acc_type`,
           `acc_capacity`, `acc_voltage`, `acc_efficiency`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `accumulator` ON `acc_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.22 – Код представления аккумуляторов

Представлние служащее для выбора всех тормозных дисков (рисунок 5.23).

```sql
CREATE OR REPLACE VIEW GET_BRAKE_DISCS AS
    SELECT `product`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           `bdi_is_ventilated`,
           GET_PRODUCT_VALUE_BY_ID(`bdi_perforation_type`) AS `bdi_perforation_type`,
           GET_PRODUCT_VALUE_BY_ID(`bdi_material`) AS `bdi_material`,
           GET_PRODUCT_VALUE_BY_ID(`bdi_installation_side`) AS `bdi_installation_side`,
           GET_PRODUCT_VALUE_BY_ID(`bdi_type`) AS `bdi_type`,
           `bdi_inner_diameter`, `bdi_outer_diameter`, `bdi_disk_width`,
           `bdi_holes_number`, `bdi_hole_width`, `fac_name`, `man_name`,
           `add_country`, `add_city`, `add_region`, `add_street`, `add_building`,
           `add_home`, `add_apartment`, `add_postcode`, `cap_width`, `cap_height`,
           `cap_length`, `cap_weight`
    FROM `product`
    JOIN `brake_disc` ON `bdi_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.23 – Код представления тормозных дисков

Представлние служащее для выбора всех тормозных колонок (рисунок 5.24).

```sql
CREATE OR REPLACE VIEW GET_BRAKE_COLUMNS AS
    SELECT `product`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           GET_PRODUCT_VALUE_BY_ID(`bco_frame_material`) AS `bco_frame_material`,
           GET_PRODUCT_VALUE_BY_ID(`bco_internal_material`) AS `bco_internal_material`,
           `bco_column_width`, `bco_column_length`, `bco_friction_coefficient`, `bco_max_load`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`,
           `add_postcode`, `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `brake_columns` ON `bco_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.24 – Код представления тормозных колонок

Представлние служащее для выбора всех зубчатых ремней (рисунок 5.25).

```sql
CREATE OR REPLACE VIEW GET_TIMING_BELTS AS
    SELECT `product`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           GET_PRODUCT_VALUE_BY_ID(`tbe_transport_type`) AS `tbe_transport_type`,
           `tbe_teeth_number`,
           GET_PRODUCT_VALUE_BY_ID(`tbe_profile_code`) AS `tbe_profile_code`,
           GET_PRODUCT_VALUE_BY_ID(`tbe_teeth_type`) AS `tbe_teeth_type`,
           `tbe_material`, `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`,
           `add_postcode`, `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `timing_belt` ON `tbe_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.25 – Код представления зубчатых ремней

Представлние служащее для выбора всех колесных дисков (рисунок 5.26).

```sql
CREATE OR REPLACE VIEW GET_WHEEL_DISCS AS
    SELECT `product`.*, `wheel_disc`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           `fac_name`, `man_name`, `add_country`, `add_city`, `add_region`,
           `add_street`, `add_building`, `add_home`, `add_apartment`,
           `add_postcode`, `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `wheel_disc` ON `wdi_id` = `pro_id`
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.26 – Код представления колесных дисков

Представлние служащее для выбора всех камер (рисунок 5.27).

```sql
CREATE OR REPLACE VIEW GET_CAMERAS AS
    SELECT `product`.*, `camera`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           `fac_name`, `man_name`, `tpr_profile_height`,
           GET_PRODUCT_VALUE_BY_ID(`tpr_type`) AS `tpr_type`,
           `tpr_diameter`, `tpr_load_index`, `tpr_speed_index`,
           `add_country`, `add_city`, `add_region`, `add_street`,
           `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `camera` ON `cam_id` = `pro_id`
    JOIN `tire_properties` USING(`tpr_id`)
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.27 – Код представления камер

Представлние служащее для выбора всех шин (рисунок 5.28).

```sql
CREATE OR REPLACE VIEW GET_TIRES AS
    SELECT `product`.*, `tire`.*, `col_name`,
           `cty_release_year`, `cty_fuel_consumption`, `cty_name`, `cty_vin_num`,
           `fac_name`, `man_name`, `tpr_profile_height`,
           GET_PRODUCT_VALUE_BY_ID(`tpr_type`) AS `tpr_type`,
           `tpr_diameter`,
           GET_PRODUCT_VALUE_BY_ID(`tpr_load_index`) AS `tpr_load_index`,
           GET_PRODUCT_VALUE_BY_ID(`tpr_speed_index`) AS `tpr_speed_index`,
           `add_country`, `add_city`, `add_region`, `add_street`,
           `add_building`, `add_home`, `add_apartment`, `add_postcode`,
           `cap_width`, `cap_height`, `cap_length`, `cap_weight`
    FROM `product`
    JOIN `tire` ON `tir_id` = `pro_id`
    JOIN `tire_properties` USING(`tpr_id`)
    JOIN `color` USING(`col_id`)
    JOIN `factory` USING(`fac_id`)
    JOIN `manufacturer` USING(`man_id`)
    JOIN `address` USING(`add_id`)
    JOIN `capacity` USING(`cap_id`)
    JOIN `car_type` USING(`cty_id`);

```

Рисунок 5.28 – Код представления шин

5.4 Триггеры базы данных

В базе данных имеется 3 триггера:
– установка времени регистрации пользователя;
– установка времени создания заказа;

– очистка статистики посещений сайта пользователями.

Триггер установки времени регистрации пользователя. Нужен для точной установки времени создания новой записи в таблице пользователей (рисунок 5.29).

```
CREATE TRIGGER `TRG_udpate_user_registration_time_before_ins`
BEFORE INSERT
ON `user`
    FOR EACH ROW
    BEGIN
        SET NEW.`use_registration_time` = CURDATE();
    END;
```

Рисунок 5.29 – Код триггера установки времени регистрации пользователя

Триггер установки времени создания заказа. Нужен для точной установки времени создания новой записи в таблице заказов (рисунок 5.30).

```
CREATE TRIGGER `TRG_udpate_order_time_created_after_ins`
BEFORE INSERT
ON `order`
    FOR EACH ROW
    BEGIN
        SET NEW.`ord_created_time` = CURDATE();
    END;
```

Рисунок 5.30 – Код триггера установки времени создания заказа

Триггер очистки статистики посещений сайта пользователями. Нужен для очистки от записей в таблице посещений при переполнении (рисунок 5.31).

```
CREATE TRIGGER `TRG_user_online_clear_before_insert`
    AFTER INSERT
    ON `user_online` FOR EACH ROW
BEGIN
    SET @count = (SELECT COUNT(*) FROM `user_online`);
    IF @count > 18446744073709551610 THEN
        CALL CLEAR_INFORMATION_ONLINE();
    END IF;
END;
```

Рисунок 5.31 – Код триггера статистики посещений сайта пользователями

6 ТЕСТИРОВАНИЕ

Тестирование проводится с целью обеспечить качество разрабатываемого программного продукта. Стандарт ISO/IEC 12207-2003, посвященный описанию систем обеспечения качества программного обеспечения, под качеством понимает совокупность характеристик программного продукта, относящихся к его способности удовлетворять установленные и предполагаемые потребности клиента.

Основным параметром качества программы является надёжность. Надёжность определяется как вероятность его работы без отказов в течении определённого периода времени, рассчитанная с учётом стоимости для пользователя каждого отказа.

Таблица 6.1 – тестирование базы данных

№ Теста	Тестируемая функциональность	Последовательность действий	Ожидаемый результат	Результат теста
1	Регистрация пользователя	Добавление кортежа в таблицу user и user_info	Успешное добавление	Тест пройден
2	Регистрация пользователя с существующим email	Добавление кортежа в таблицу user_info с существующим email	Запрет создания	Тест пройден
3	Добавление склада	Добавление кортежа в таблицу warehouse	Успешное добавление	Тест пройден
4	Добавление пункта самовываоза	Добавление кортежа в таблицу reception_point	Успешное добавление	Тест пройден
5	Добавление транспорта	Добавление кортежа в таблицу car	Успешное добавление	Тест пройден
6	Добавление шин	Добавление кортежа в таблицу product и tire	Успешное добавление	Тест пройден
7	Создание нового пользователя с помощью функции CREATE_USER	Добавление кортежа в таблицу user и user_info	Успешное добавление	Тест пройден
8	Обновления информации с помощью функции UPDATE_USER	Добавление кортежа в таблицу user_info, обновление uin_id в таблице user	Успешное добавление	Тест пройден

Продолжение таблицы 6.1

№ Теста	Тестируемая функциональность	Последовательность действий	Ожидаемый результат	Результат теста
9	Добавление адреса	Добавление кортежа в таблицу address	Успешное добавление	Тест пройден
10	Добавление скидки на продукт	Добавление кортежа в таблицу discount и m2m_product_discount	Успешное добавление	Тест пройден
11	Добавление нового предприятия	Добавление кортежа в таблицу factory	Успешное добавление	Тест пройден
12	Удаление пользователя	Удаление пользователя + каскадное удаление, связанных с ним кортежей	Удаление пользователя и его зависимых кортежей	Тест пройден
13	Выполнение функции GET_ID_BY_PRODUCT_VALUE	Вызов соответствующей функции	Получения id	Тест пройден

ЗАКЛЮЧЕНИЕ

Развитие аппаратных и программных технологий привело к большой популярности сети Интернет и позволило ей занять лидирующее положение среди основных инструментов ведения бизнеса, в частности, электронной торговли. Присутствие торговой компании в Интернете необходимо не только с целью получения и наращивания желаемой прибыли, но и для успешной конкурентной борьбы в современных условиях.

Онлайн-магазин – наиболее популярный вид онлайновой торговли. В процессе создания данного веб-ресурса важно грамотно выстроить стратегию ведения бизнеса. В ряд важнейших вопросов, которые предстоит решить торговой компании, входят разработка организационной структуры и ассортиментной политики, выбор способа построения и дальнейшего сопровождения информационной системы интернет-магазина, организация службы доставки, маркетинговая деятельность и, что немаловажно, разработка качественного веб-дизайна предоставляемого ресурса.

В начале разработки я ставил перед собой цели:

− создать качественную базу данных;

− получить опыт написания баз данных;

− изучить различные подходы к проектированию реляционных баз данных;

− закрепить знания различных аспектов СУБД MySQL;

− реализовать возможность последующего совершенствования базы данных в зависимости от требований потребителя;

− получить опыт проектирования бизнес-логики.

В ходе работы мною решалось множество проблем связанных с реализацией моих идей, что позволило приобрести ценный опыт. Из главных уроков данного курсового проекта хотелось бы выделить:

− закрепление знаний о СУБД MySQL;

− огромный опыт в проектировании баз данных;

− опыт в обработке событий.

Вследствие этих факторов мной был разработан качественная база данных с большими перспективами совершенствования и развития. Также мной были получены ценные знания, которые пригодятся мне в будущем.

СПИСОК ЛИТЕРАТУРЫ

[1] Куликов Святослав Святославович «Реляционные базы данных в примерах» (2021) – 422 с.

[2] Куликов Святослав Святославович «Работа с MySQL, MSSQL Server и Oracle» (2021) – 602 с.

[3] https://tl24.by/about [Электронный портал]. Интернет-магазин

[4] https://dvmparts.by/about [Электронный портал]. Интернет-магазин

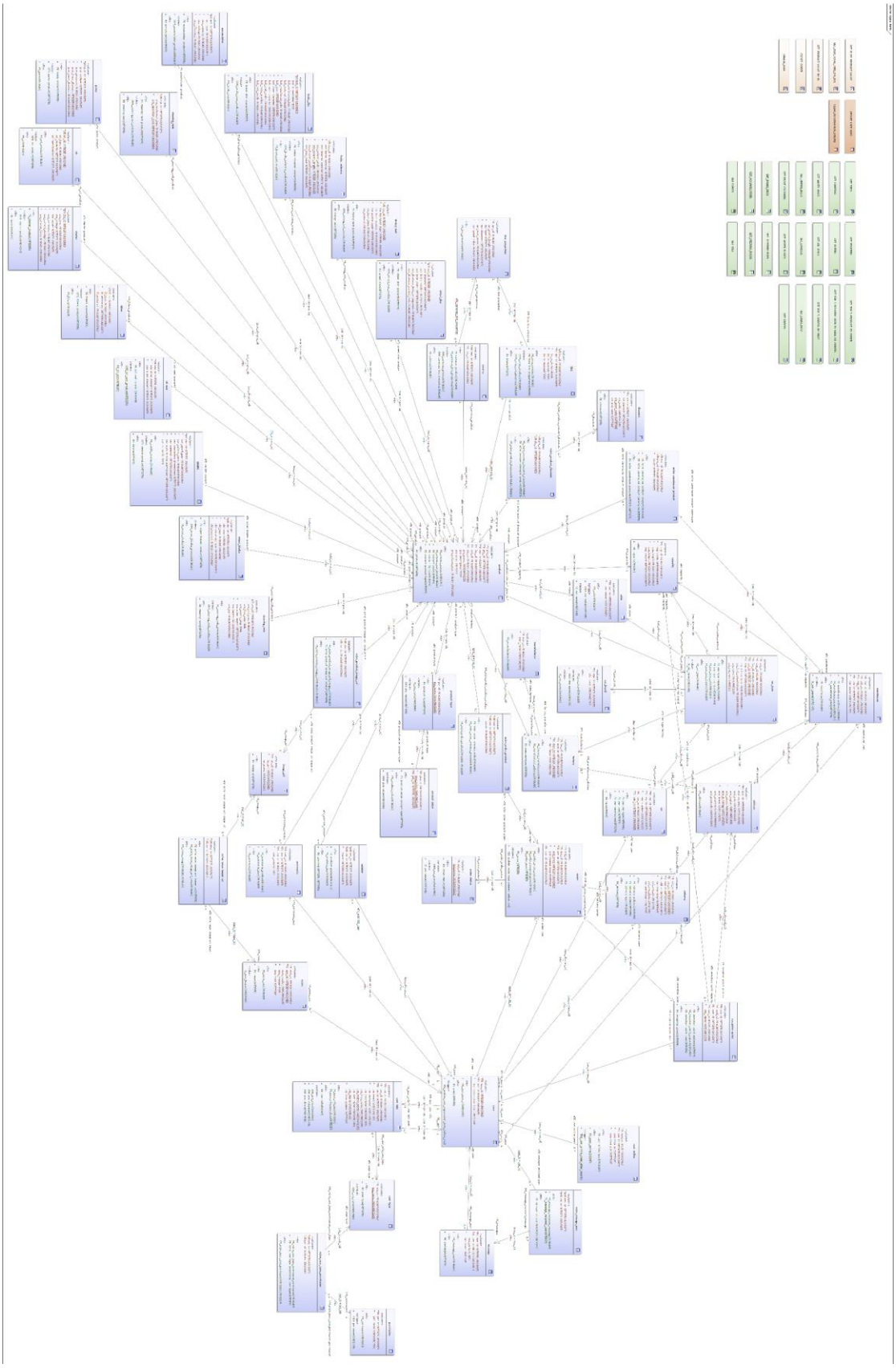
[5] https://byavto.by/about [Электронный портал]. Интернет-магазин

[6] https://ru.wikipedia.org/wiki/Интернет-магазин [Электронный портал]. Интернет-магазин

[7] https://web-creator.ru/articles/mysql [Электронный портал]. MySQL — система управления базами данных

ПРИЛОЖЕНИЕ

Физическая модель БД



Обозначение	Наименование	Дополнительные сведения
	Текстовые документы	
БГУИР КП 1–40 01 01 603 ПЗ	Пояснительная записка	67 с.
	Графические документы	
ГУИР 851006 603 ПД	"Физическая модель БД", А1, плакат	Формат А1

					БГУИР КП 1-40 01 01 603 Д1		
Изм.	Л.	№ докум.	Подп.	Дата	«Онлайн-магазин автомобильных запчастей» Ведомость курсовой работы	Лист	Листов
Разраб.		Верещагин Н.В.				Т 67	67
Пров.		Фадеева Е.Е.				Кафедра ПОИТ гр. 851006	