*Федеральное государственное бюджетное образовательное учреждение высшего образования*

*«Владимирский государственный университет*

*имени Александра Григорьевича и Николая Григорьевича Столетовых»*

*Кафедра информационных систем и программной инженерии*

***КУРСОВОЙ ПРОЕКТ***

*Разработка структуры базы данных для*

*информационной системы «Производство»*

*Выполнил: студент гр. ПРИ-120*

*Грачев Д. А.*

*Принял: доц. Вершинин В.В.*

*Владимир, 2022*

Оглавление

[2 Проектирование БД 3](#_Toc102674024)

[Этап 2 4](#_Toc102674025)

[Этап 3 6](#_Toc102674026)

[Этап 4 7](#_Toc102674027)

[ПРИЛОЖЕНИЕ А 8](#_Toc102674028)

[ПРИЛОЖЕНИЕ Б 9](#_Toc102674029)

## ПРОЕКТИРОВАНИЕ БД

* 1. **Цель и задачи, решаемые подсистемой хранения данных**

Цель работы: разработать подсистему хранения данных, которая будет позволять сохранять фактические данные о производстве грузовых автомобилей, их поставки заказчикам и наличия требуемых деталей у поставщиков.

Для достижения поставленной цели необходимо решить следующие задачи:

* хранение списка автомобилей;
* хранение списка производителей компонентов;
* отметка о доставке техники;
  1. **Описание предметной области**

На заводе, производящем грузовые автомобили, важно отслеживать выполнение заказа, его доставку и сборку. В БД хранится информация о различных моделях техники и их составных компонентов, так же хранится информация о доставках и о каждом произведенном грузовике отдельно

* 1. **Пользователи разрабатываемой подсистемы**

*Менеджер по заказам* – сотрудник завода, может просматривать доставки, добавлять их, изменять статус доставки, редактировать таблицу с информацией об автомобилях.

*Доставщик* – сотрудник завода, может просматривать заказы, изменять статус заказа, делать выборки по адресам доставки, по компаниям заказчикам

*Инженер –* сотрудник завода, может просматривать таблицу моделей и создавать новые модели

*Рабочий* – сотрудник завода, может просматривать таблицы с моделями и заказами, делать выборки по моделям

*Заказчик –* компания, заказывающая технику, может просматривать состояние заказа

* 1. **Начальная оценка и выделение сущностей**

Для предметной области «Производство» были выделены следующие сущности: автомобиль, компания, модель, корпус, двигатель, доставка.

* 1. **Словарь предметной области;**

*Автомобиль –* единица производимого товара.

Атрибуты: номер, номер модели, номер заказа, год выпуска

*Модель –* вид единицы техники.

Атрибуты: номер модели, название, номер корпуса, номер мотора, объем выпуска

*Доставка –* информация о доставке.

Атрибуты: номер доставки, заказчик, адрес доставки, телефон, статус доставки

*Состояние доставки –* информация о статусе доставки, может принимать 3 состояния: Изготавливается, В пути, Доставлен.

*Корпус –* информация о типе корпуса.

Атрибуты: номер корпуса, название, размер, компания изготовитель

*Двигатель –* информация о типе двигателе.

Атрибуты: номер мотора, название, мощность, компания изготовитель.

## Этап 2

Нормализация — это процесс организации данных в базе данных, включающий создание таблиц и установление отношений между ними в соответствии с правилами, которые обеспечивают защиту данных и делают базу данных более гибкой, устраняя избыточность и несогласованные зависимости.

**Первая нормальная форма**

* Отсутствуют повторяющиеся группы в отдельных таблицах
* Отдельная таблица для каждого набора связанных данных
* Каждый набор данных идентифицируется с помощью первичного ключа

**Вторая нормальная форма**

* Наличие отдельных таблиц для наборов значений, относящимся к нескольким записям
* Таблицы связаны с помощью внешнего ключа

**Третья нормальная форма**

* Устранены поля, не зависящие от ключа

**Результат применения аппарата нормализации**

В результате применения аппарата нормализации была достигнута 3 нормальная форма базы данных: в таблицах отсутствуют повторяющиеся строки; каждая таблица идентифицируется с помощью первичного ключа; таблица связаны с помощью внешнего ключа; отсутствуют поля, не зависящие от ключа.

**Логическая и физическая схема базы данных представлены в приложениях А и Б соответственно**

**Описание связей между сущностями БД**

|  |  |  |  |
| --- | --- | --- | --- |
| Связь | Обязательность | Кратность | Идентифицируемость |
| Таблица «Автомобили» и таблица «Модель» | Связь обязательна, так как у автомобиля есть модель | Один-ко-многим  Автомобиль имеет одну модель  Много автомобилей могут быть одной модели | Связь не идентифицируемая, так как первичный ключ в таблице «Автомобили» не является частью первичного ключа в таблице «Модель» |
| Таблица «Автомобили» и таблица «Доставка» | Связь обязательна, так как автомобиль производится для заказа | Один-ко-многим  Автомобиль может быть в одном заказе  В заказе может быть несколько автомобилей | Связь не идентифицируемая, так как первичный ключ в таблице «Автомобили» не является частью первичного ключа в таблице «Заказ» |
| Таблица «Модель» и таблица «Корпус» | Связь обязательна, так как у модель присутствует корпус | Один-ко-многим  У модели может быть один корпус  С одним корпусом может быть несколько моделей | Связь не идентифицируемая, так как первичный ключ в таблице «Модель» не является частью первичного ключа в таблице «Корпус» |
| Таблица «Модель» и таблица «Двигатель» | Связь обязательна, так как у модели присутствует двигатель | Один-ко-многим  У модели может быть один двигатель  С одним двигателем могут быть разные модели | Связь не идентифицируемая, так как первичный ключ в таблице «Модель» не является частью первичного ключа в таблице «Двигатель» |
| Таблица «Доставка» и таблица «Компания» | Связь обязательна, так как доставку заказывает определенная компания | Один-ко-многим  Компания может сделать много заказов  Один заказ может ассоциироваться с одной кампанией | Связь не идентифицируемая, так как первичный ключ в таблице «Компания» не является частью первичного ключа в таблице «Доставка» |
| Таблица «Корпус» и таблица «Компания» | Связь обязательна, так как у корпуса есть компания-производитель | Один-ко-многим  Компания может производить несколько видов корпусов  Один корпус может быть произведен одной компанией | Связь не идентифицируемая, так как первичный ключ в таблице «Корпус» не является частью первичного ключа в таблице «Компания» |
| Таблица «Двигатель» и таблица «Компания» | Связь обязательна, так как у двигателя есть компания-производитель | Один-ко-многим  Компания может производить несколько видов двигателей  Один двигатель может быть произведен одной компанией | Связь не идентифицируемая, так как первичный ключ в таблице «Двигатель» не является частью первичного ключа в таблице «Компания» |

## Этап 3

**Возможный функционал проектируемой БД:**

1. **Функция DeliveryTime**

Принимает в аргументы *ID автомобиля* и возвращает срок его доставки, в случае если автомобиль еще не доставлен – возвращает -1

1. **Функция ReleaseTime**

Принимает в аргументы *ID модели* и возвращает время её производства путем высчитывания разницы между последним годом выпуска данной модели и первым

1. **Функция FixDelivery**

Не принимает аргументов, с помощью **курсора deliveryCursor** проходит по таблице Delivery и проставляет статус «Доставлено» у строк, где есть дата доставки, но неправильно стоит статус, возвращает строку с ID исправленных строк в виде «ID-ID-ID»

1. **Процедура CreateAuto**

Принимает на вход *ID модели, Покупателя, Адрес и ID доставки,* добавляет запись в таблицу Auto, если *ID доставки* был указан «-1» или не указан создает новую доставку для этого автомобиля, оформленную на *Покупателя*, если *ID доставки* передается, то привязывает новый автомобиль к этой доставке

1. **Процедура SwitchDeliveryStatus**

Принимает на вход *ID заказа* и меняет его статус:

* 1. Если *Статус* был «Изготавливается», то меняет его на «В пути», а также уменьшает *Количество двигателей и корпусов*, которые были использованы в производстве автомобилей из этого заказа
  2. Если *Статус* был «В пути», то меняет его на «Доставлен»

1. **Процедура AutoArchiving**

Принимает на вход  *Год* и архивирует автомобили, произведенные до заданного года, удаляя их из таблицы Auto и перенося в таблицу AutoArchive

1. **Представление AutoWithComponent**

Предоставляет таблицу с информацией о автомобиле: *ID автомобиля,* *Названием модели, Название двигателя, Мощность двигателя, Название корпуса, Размеры корпуса* из таблиц Auto, Model**,** EngineиBody

1. **Триггер OnDelivered**

При изменении *статуса*  заказа в таблице Orderна «Доставлен», у этого заказа проставляется дата доставки, равная сегодняшнему дню

1. **Триггер PlusModelCount**

При добавлении записи в таблицу Autoувеличивается *Объем выпуска* модели в таблице Model

## Реализация компонентов базы данных

**Реализация компонентов базы данных**

**Физическая структура базы данных**

**Таблицы БД**

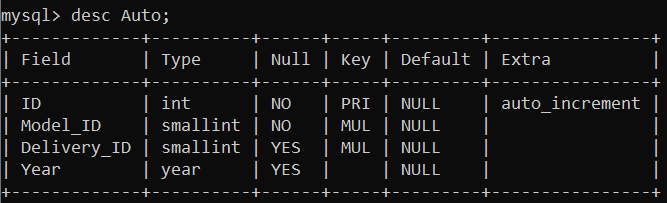


Рисунок 1. Структура таблицы Auto

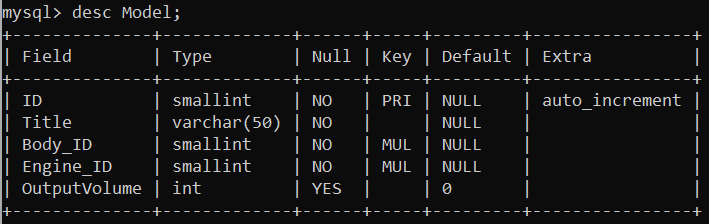


Рисунок 2. Структура таблицы Model

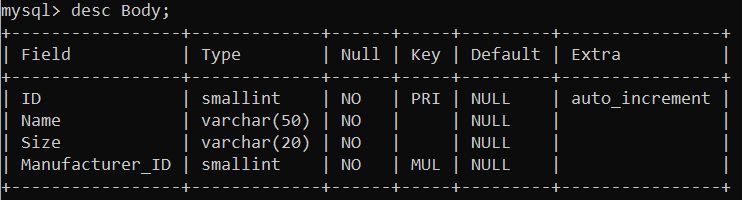


Рисунок 3. Структура таблицы Body

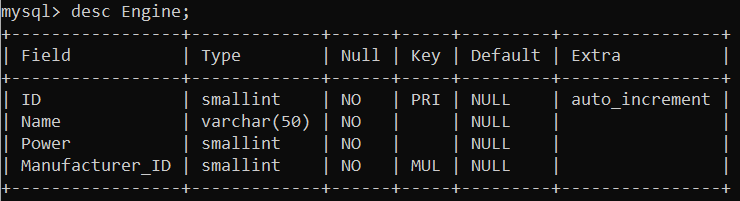


Рисунок 4. Структура таблицы Engine

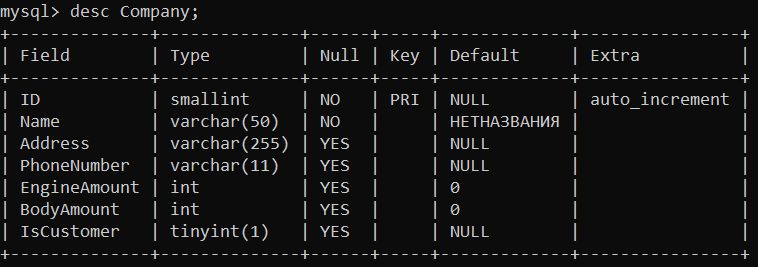


Рисунок 5. Структура таблицы Company

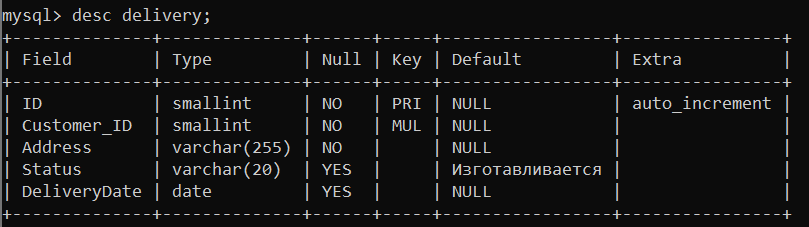


Рисунок 6. Структура таблицы Delivery

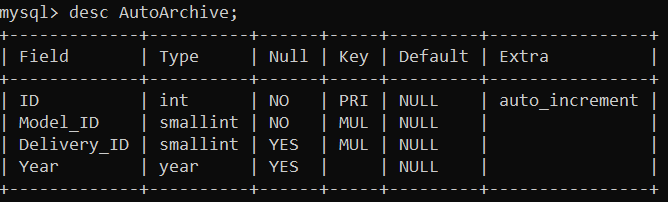


Рисунок 7. Структура таблицы AutoArchive

**Спецификация функций:**

1. **Функция DeliveryTime()**

CREATE FUNCTION DeliveryTime(AutoID SMALLINT) RETURNS SMALLINT DETERMINISTIC

BEGIN

DECLARE ReleaseYear SMALLINT;

DECLARE DeliveryID SMALLINT;

DECLARE DeliveryYear SMALLINT;

SELECT Year,Delivery\_ID from

(SELECT ID,Delivery\_ID,Year FROM Auto

UNION

SELECT ID,Delivery\_ID,Year FROM AutoArchive)

AS AllAuto

WHERE ID = AutoID

INTO ReleaseYear,DeliveryID;

SELECT YEAR(DeliveryDate) from Delivery

WHERE ID = DeliveryID

INTO DeliveryYear;

IF DeliveryYear IS NULL THEN RETURN -1;

ELSE RETURN (DeliveryYear - ReleaseYear + 1);

END IF;

END$$

1. **Функция ReleaseTime()**

CREATE FUNCTION ReleaseTime(ModelID SMALLINT) RETURNS SMALLINT DETERMINISTIC

BEGIN

DECLARE MinYear SMALLINT;

DECLARE MaxYear SMALLINT;

SELECT MIN(Year) FROM

(SELECT Model\_ID,Year FROM Auto

UNION

SELECT Model\_ID,Year FROM AutoArchive)

AS AllAuto

WHERE Model\_ID = ModelID

INTO MinYear;

SELECT MAX(Year) FROM

(SELECT Model\_ID,Year FROM Auto

UNION

SELECT Model\_ID,Year FROM AutoArchive)

AS AllAuto

WHERE Model\_ID = ModelID

INTO MaxYear;

RETURN (MaxYear - MinYear + 1);

END$$

1. **Функция FixDelivery()**

CREATE FUNCTION FixDelivery() RETURNS VARCHAR(255) DETERMINISTIC

BEGIN

DECLARE done INT DEFAULT 0;

DECLARE DelID SMALLINT;

DECLARE DelStatus VARCHAR(20);

DECLARE DelDate DATE;

DECLARE ErrCounter VARCHAR(255);

DECLARE deliveryCursor CURSOR

FOR SELECT ID,Status,DeliveryDate

FROM Delivery;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

OPEN deliveryCursor;

featchLoop: WHILE done = 0 DO

FETCH deliveryCursor INTO DelID,DelStatus,DelDate;

IF DelStatus != "Доставлен" AND DelDate IS NOT NULL THEN

UPDATE Delivery

SET Status = "Доставлен"

WHERE ID = DelID;

SET ErrCounter = CONCAT\_WS('-',ErrCounter,DelID);

END IF;

END WHILE featchLoop;

CLOSE deliveryCursor;

RETURN LEFT(ErrCounter,LENGTH(ErrCounter)-3);

END$$

**Спецификация процедур:**

1. **Процедура CreateAuto()**

CREATE PROCEDURE CreateAuto(

IN ModelID SMALLINT,

IN Customer VARCHAR(50),

IN Address VARCHAR(255),

IN DeliveryID SMALLINT )

BEGIN

IF DeliveryID = -1 OR DeliveryID IS NULL THEN

SELECT ID FROM Company

WHERE Name = Customer

INTO @delivery\_company\_ID;

INSERT Delivery(Customer\_ID,Address)

VALUES (@delivery\_company\_ID, Address);

SELECT max(ID) FROM Delivery

INTO DeliveryID;

END IF;

INSERT Auto(Model\_ID,Delivery\_ID,Year)

VALUES (ModelID,DeliveryID,YEAR(NOW()));

END$$

1. **Процедура SwitchDeliveryStatus()**

CREATE PROCEDURE SwitchDeliveryStatus(IN DeliveryID SMALLINT)

BEGIN

SELECT Status FROM Delivery

WHERE ID = DeliveryID

INTO @usedStatus;

IF @usedStatus = "Изготавливается" THEN

UPDATE Delivery

SET Status = "В пути"

WHERE Delivery.ID = DeliveryID;

UPDATE Company

JOIN Engine ON Company.ID = Engine.Manufacturer\_ID

JOIN Model ON Engine.ID = Model.Engine\_ID

JOIN Auto ON Model.ID = Auto.Model\_ID

SET EngineAmount = EngineAmount - 1

WHERE Auto.Delivery\_ID = DeliveryID;

UPDATE Company

JOIN Body ON Company.ID = Body.Manufacturer\_ID

JOIN Model ON Body.ID = Model.Body\_ID

JOIN Auto ON Model.ID = Auto.Model\_ID

SET BodyAmount = BodyAmount - 1

WHERE Auto.Delivery\_ID = DeliveryID;

ELSEIF @usedStatus = "В пути" THEN

UPDATE Delivery

SET Status = "Доставлен"

WHERE Delivery.ID = DeliveryID;

END IF;

END$$

1. **Процедура AutoArchiving()**

CREATE PROCEDURE AutoArchiving(IN YearOfIssue SMALLINT)

BEGIN

INSERT INTO AutoArchive

SELECT \* from Auto

WHERE Auto.Year <= YearOfIssue;

DELETE FROM Auto

WHERE Auto.Year <= YearOfIssue;

END$$

**Спецификация триггеров:**

1. **Триггер OnDelivered**

CREATE TRIGGER OnDelivered BEFORE UPDATE ON Delivery

FOR EACH ROW

IF NEW.Status="Доставлен" AND NEW.DeliveryDate IS NULL THEN

SET NEW.DeliveryDate = CURDATE();

END IF$$

1. **Триггер PlusModelCount**

CREATE TRIGGER PlusModelCount BEFORE INSERT ON Auto

FOR EACH ROW

UPDATE Model

SET OutputVolume = OutputVolume + 1

WHERE Model.ID = NEW.Model\_ID;

**Спецификация представления:**

1. **Представление** **AutoWithComponent**

CREATE VIEW AutoWithComponent AS

SELECT Auto.id AS AutoID,

Model.Title AS Model,

Engine.Name AS Engine,

Engine.Power AS Power,

Body.Name AS Body,

Body.Size AS Size

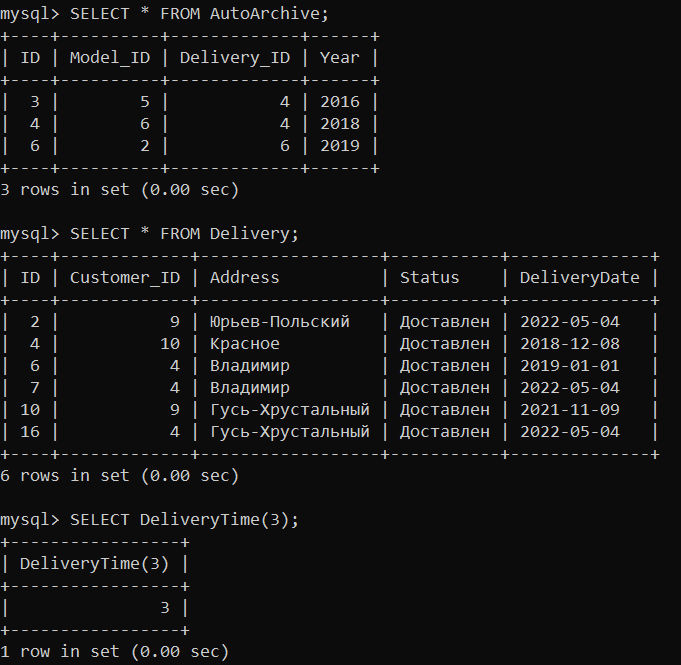
FROM Auto JOIN Model ON Auto.Model\_ID=Model.ID

JOIN Engine ON Model.Engine\_ID=Engine.ID

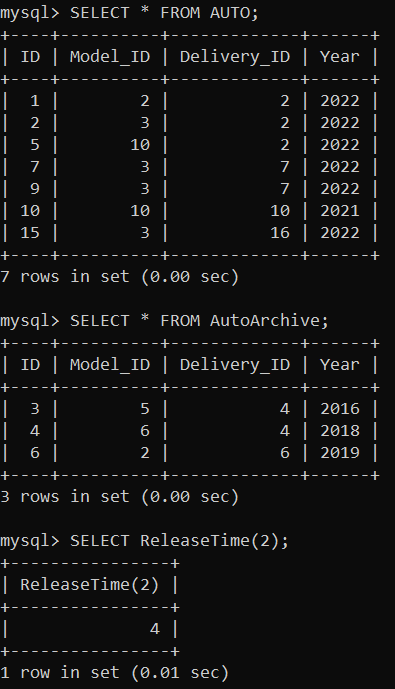
JOIN Body ON Model.Body\_ID=Body.ID;

**Тестирование системы:**

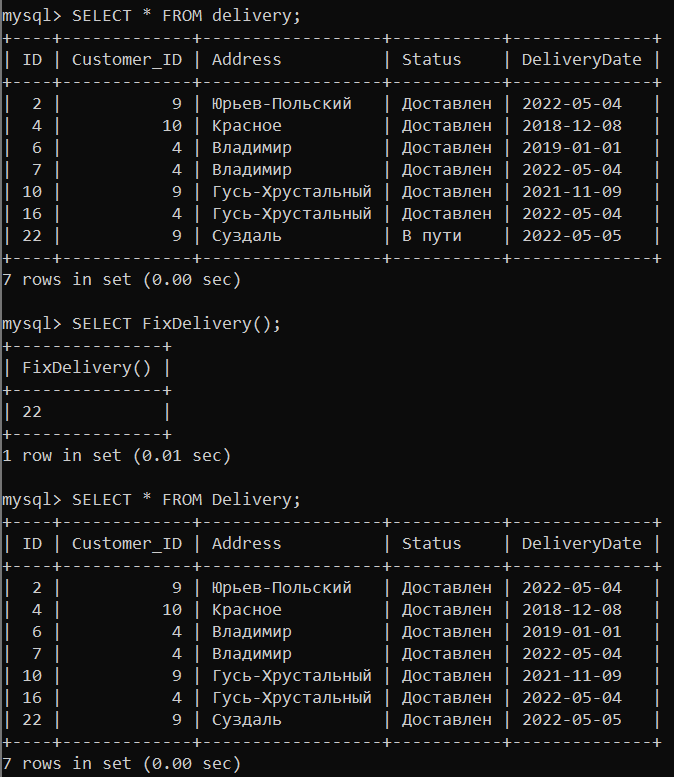
1. **Функция DeliveryTime(3)**



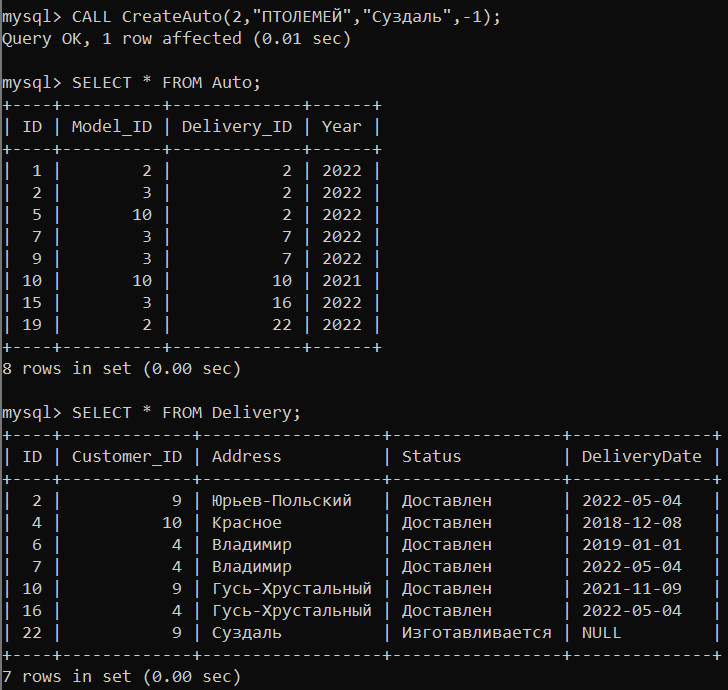
1. **Функция ReleaseTime()**



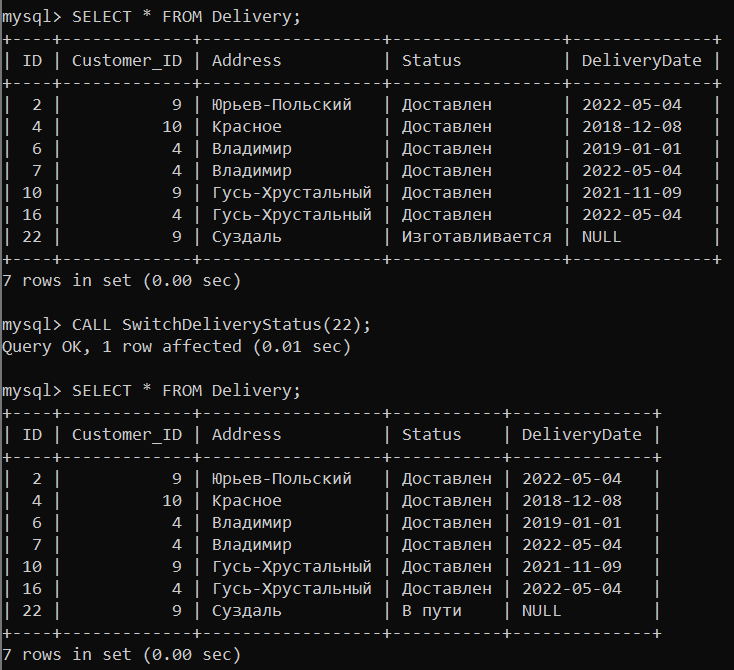
1. **Функция FixDelivery()**



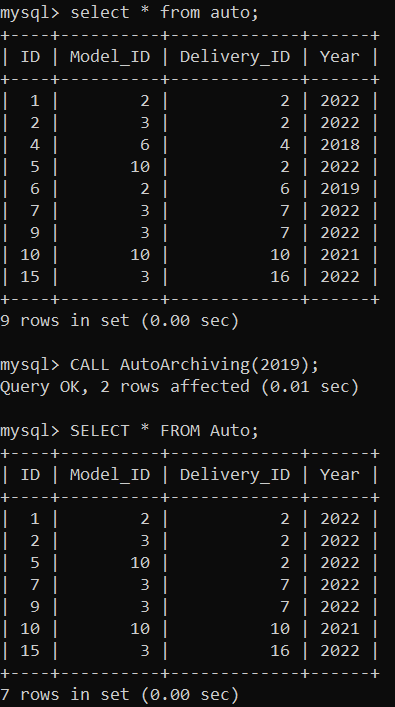
1. **Процедура CreateAuto(2,”ПИЛОМЕЙ”,”Суздаль”,-1)**

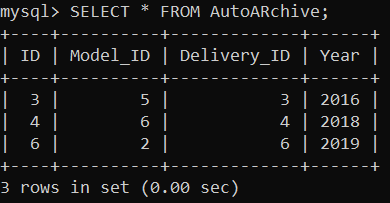


1. **Процедура SwitchDeliveryStatus(22)**

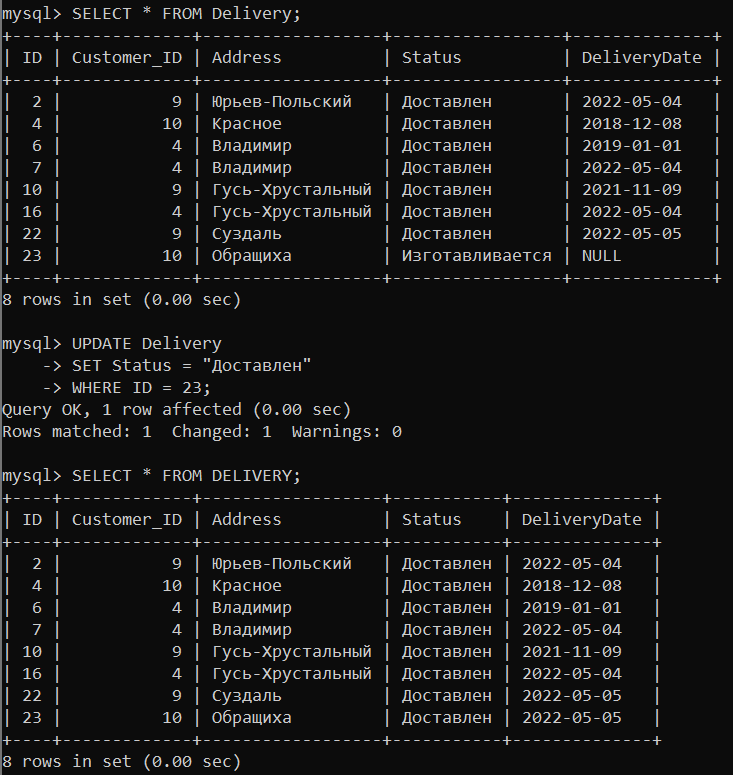


1. **Процедура AutoArchiving(2019)**

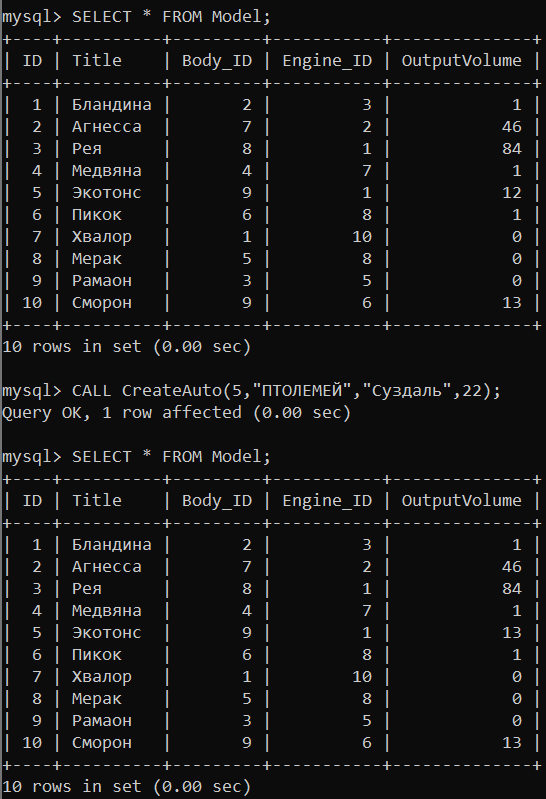




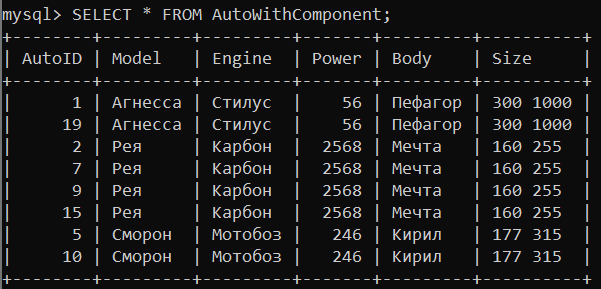
1. **Триггер OnDelivered**



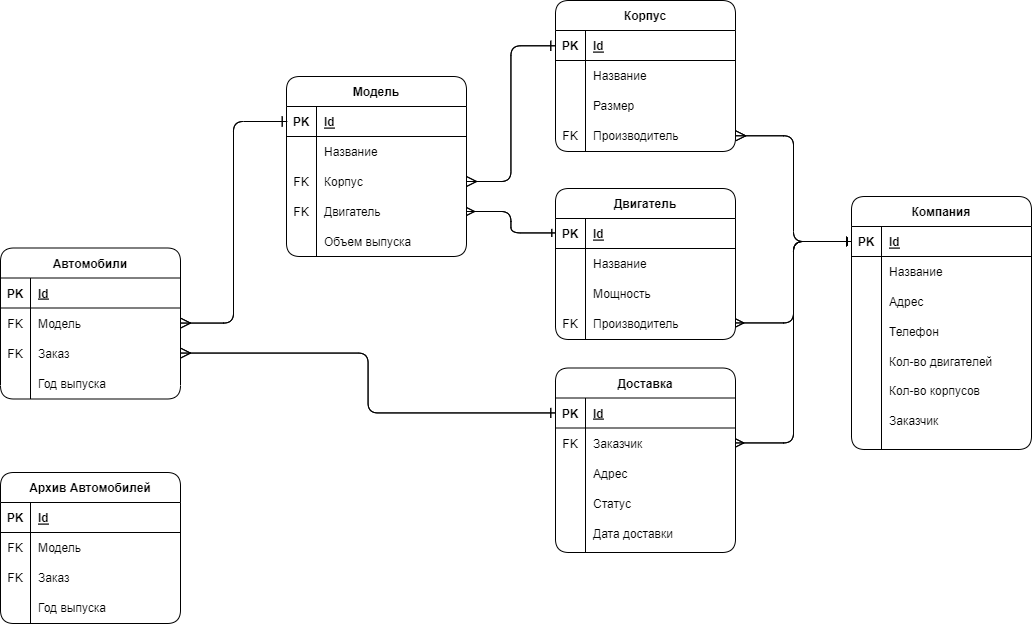
1. **Триггер PlusModelCount**



1. **Представление AutoWithComponent**

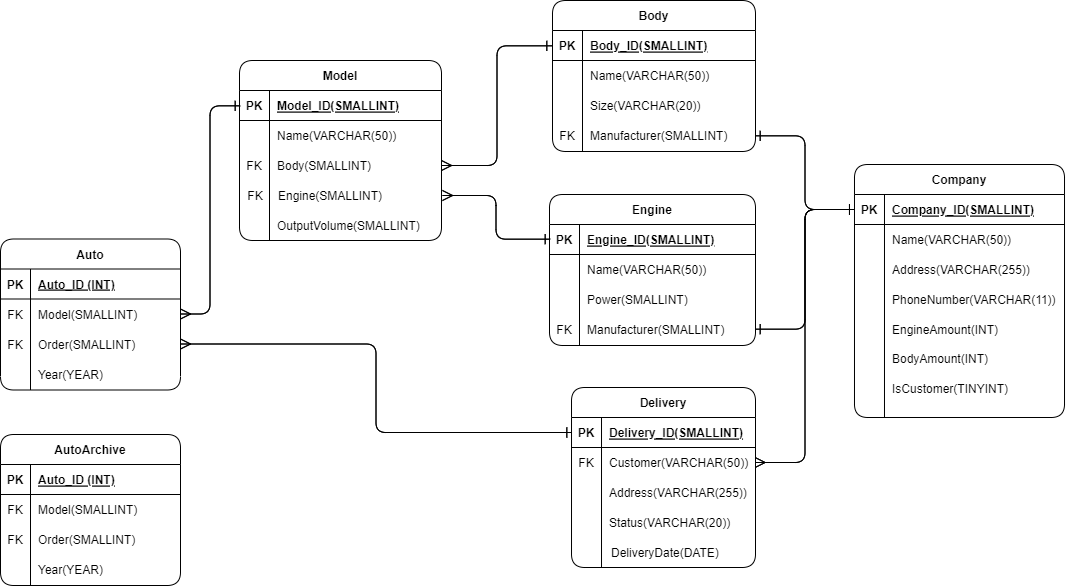


## ПРИЛОЖЕНИЕ А



Логическая схема БД

## ПРИЛОЖЕНИЕ Б



Физическая схема БД