МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

Факультет прикладной математики, информатики и механики
Кафедра математического обеспечения ЭВМ

Разработка структуры базы данных автосервиса «Автокомплект».

Отчет по дисциплине «Реляционные СУБД»

02.03.02 Фундаментальная информатика и информационные технологии

Зав. Кафедрой ___ д.т.н. Абрамов Г. В
Обучающийся ___ Вафин А.Р.
Руководитель ___ к.ф.-м.н. Астахова И. Ф.

Воронеж 2025

Описание базы данных

База данных автосервиса «Автокомплект» разработана для учёта и управления всеми аспектами деятельности сервисного центра.
Основная цель — обеспечение целостного, непротиворечивого и полного хранения информации об автомобилях, их владельцах, проведённых ремонтах, используемых запчастях и взаимодействии с поставщиками.

База данных состоит из следующих основных сущностей:

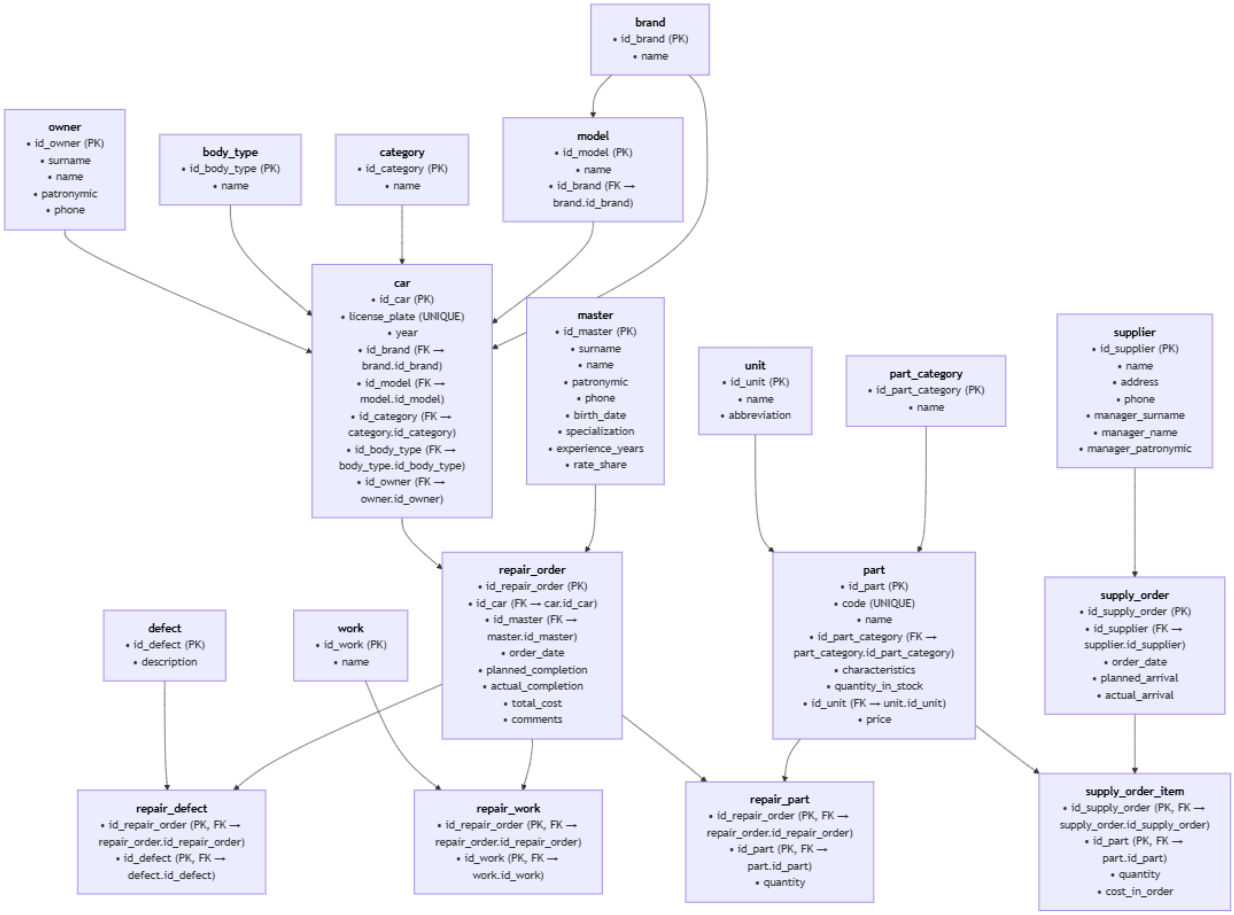
- Справочники:
    - brand — марки автомобилей;
    - model — модели автомобилей (связаны с марками);
    - category — категории автомобилей («Легковой», «Внедорожник» и др.);
    - body_type — типы кузовов («Седан», «Хэтчбек» и др.);
    - unit — единицы измерения запчастей («штука», «комплект»);
    - part_category — категории запчастей («Тормозная система», «Подвеска», «Электрика»).
- Владельцы и автомобили:
    - Таблица owner хранит ФИО и телефон владельца.
    - Таблица car содержит госномер, год выпуска, ссылки на марку, модель, категорию, кузов и владельца.
    - У одного владельца может быть несколько автомобилей.
- Мастера:
    - Таблица master включает ФИО, контактные данные, дату рождения, специализацию, стаж и долю ставки.
- Запчасти и поставки:
    - Таблица part описывает запчасти: артикул, название, категорию, характеристики, остаток, единицу измерения и цену.
    - Поставщики (supplier) характеризуются названием, адресом, телефоном и данными менеджера.
    - Заказы поставок (supply_order) фиксируют дату заказа, планируемое и фактическое поступление.

    - Состав заказа хранится в supply_order_item (запчасть, количество, стоимость позиции).

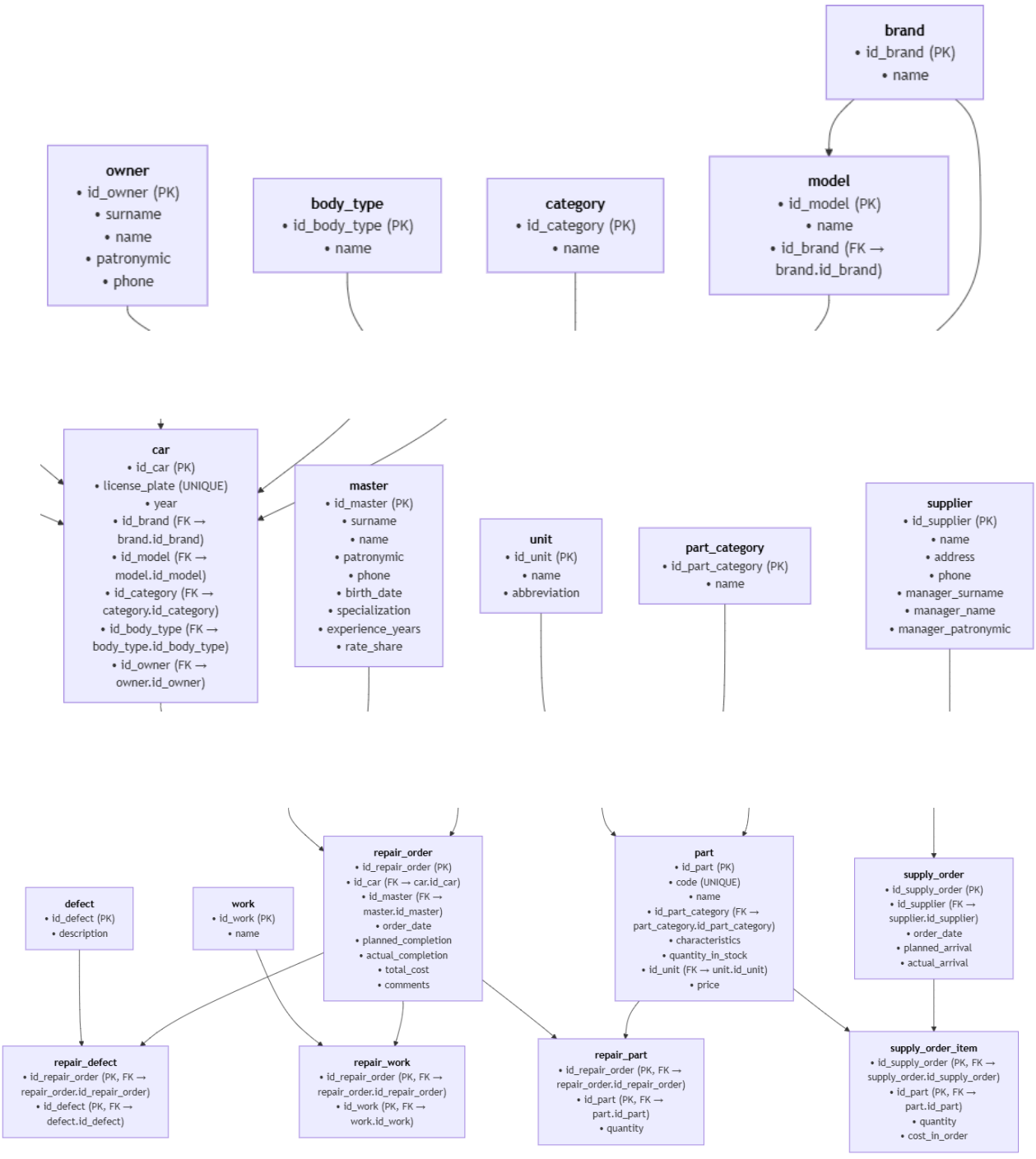
- Ремонтные заказы:
    - Каждый заказ (repair_order) привязан к автомобилю и мастеру, содержит дату поступления, плановый и фактический сроки, общую стоимость и комментарий.
    - Неисправности (defect) и виды работ (work) вынесены в отдельные справочники.
    - Связи «заказ–неисправность», «заказ–работа» и «заказ–запчасть» реализованы через промежуточные таблицы: repair_defect, repair_work, repair_part.

Все связи между таблицами обеспечены внешними ключами, что гарантирует ссылочную целостность данных.
Уникальные ограничения применяются к госномерам, артикулам запчастей и другим идентификаторам, исключающим дублирование.
База данных полностью соответствует требованиям предметной области автосервиса и поддерживает выполнение всех аналитических и учётных задач.

Схема БД


brand
• id_brand (PK)
• name
owner
• id_owner (PK)
• surname
• name
• patronymic
• phone
body_type
• id_body_type (PK)
• name
category
• id_category (PK)
• name
model
• id_model (PK)
• name
• id_brand (FK → brand.id_brand)
car
• id_car (PK)
• license_plate (UNIQUE)
• year
• id_brand (FK → brand.id_brand)
• id_model (FK → model.id_model)
• id_category (FK → category.id_category)
• id_body_type (FK → body_type.id_body_type)
• id_owner (FK → owner.id_owner)
master
• id_master (PK)
• surname
• name
• patronymic
• phone
• birth_date
• specialization
• experience_years
• rate_share
unit
• id_unit (PK)
• name
• abbreviation
part_category
• id_part_category (PK)
• name
supplier
• id_supplier (PK)
• name
• address
• phone
• manager_surname
• manager_name
• manager_patronymic
repair_order
• id_repair_order (PK)
• id_car (FK → car.id_car)
• id_master (FK → master.id_master)
• order_date
• planned_completion
• actual_completion
• total_cost
• comments
part
• id_part (PK)
• code (UNIQUE)
• name
• id_part_category (FK → part_category.id_part_category)
• characteristics
• quantity_in_stock
• id_unit (FK → unit.id_unit)
• price
supply_order
• id_supply_order (PK)
• id_supplier (FK → supplier.id_supplier)
• order_date
• planned_arrival
• actual_arrival
defect
• id_defect (PK)
• description
work
• id_work (PK)
• name
repair_defect
• id_repair_order (PK, FK → repair_order.id_repair_order)
• id_defect (PK, FK → defect.id_defect)
repair_work
• id_repair_order (PK, FK → repair_order.id_repair_order)
• id_work (PK, FK → work.id_work)
repair_part
• id_repair_order (PK, FK → repair_order.id_repair_order)
• id_part (PK, FK → part.id_part)
• quantity
supply_order_item
• id_supply_order (PK, FK → supply_order.id_supply_order)
• id_part (PK, FK → part.id_part)
• quantity
• cost_in_order

Увеличенное:



Создание и заполнение базы данных

Для реализации проектной модели в СУБД Oracle были подготовлены SQL-скрипты, содержащие операторы CREATE TABLE для создания структуры базы данных и INSERT INTO для наполнения таблиц тестовыми данными.

Создание таблиц выполнялось с учётом всех определённых ранее связей и ограничений целостности: первичных и внешних ключей, уникальности, проверок типов данных и диапазонов значений.

После создания структуры база данных была заполнена тестовыми записями, обеспечивающими корректную работу запросов и отчётов. Объём данных подобран таким образом, чтобы каждая таблица содержала несколько десятков строк, а результаты контрольных запросов были непустыми и демонстрационными.

Далее по тексту приведены фрагменты скриптов:

- определения таблиц (CREATE TABLE ...),
- вставки данных (INSERT INTO ... VALUES ...),
- дроп таблиц (BEGIN EXECUTE … TABLE …).

1. Создание БД

```
-- === Безопасный дроп таблиц для БД автосервиса «Автокомплект» ===


-- Сначала удаляем таблицы, связанные с repair_order (потомки)

BEGIN EXECUTE IMMEDIATE 'DROP TABLE repair_defect PURGE';
EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE;
END IF; END;

/

BEGIN EXECUTE IMMEDIATE 'DROP TABLE repair_work PURGE';
EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE;
END IF; END;

/

BEGIN EXECUTE IMMEDIATE 'DROP TABLE repair_part PURGE';
EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE;
END IF; END;

/
```

```
-- Затем основные зависимые таблицы

BEGIN EXECUTE IMMEDIATE 'DROP TABLE repair_order PURGE';
EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE;
END IF; END;

/

BEGIN EXECUTE IMMEDIATE 'DROP TABLE supply_order_item PURGE';
EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE;
END IF; END;

/

BEGIN EXECUTE IMMEDIATE 'DROP TABLE supply_order PURGE';
EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE;
END IF; END;

/


-- Далее — основные сущности

BEGIN EXECUTE IMMEDIATE 'DROP TABLE car PURGE'; EXCEPTION
WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;

/

BEGIN EXECUTE IMMEDIATE 'DROP TABLE master PURGE'; EXCEPTION
WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;

/

BEGIN EXECUTE IMMEDIATE 'DROP TABLE owner PURGE'; EXCEPTION
WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;

/

BEGIN EXECUTE IMMEDIATE 'DROP TABLE part PURGE'; EXCEPTION
WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;

/

BEGIN EXECUTE IMMEDIATE 'DROP TABLE supplier PURGE';
EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE;
END IF; END;

/
```

```
-- Справочники (в любом порядке, но после зависимых)
BEGIN EXECUTE IMMEDIATE 'DROP TABLE model PURGE'; EXCEPTION
WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;
/
BEGIN EXECUTE IMMEDIATE 'DROP TABLE brand PURGE'; EXCEPTION
WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;
/
BEGIN EXECUTE IMMEDIATE 'DROP TABLE category PURGE';
EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE;
END IF; END;
/
BEGIN EXECUTE IMMEDIATE 'DROP TABLE body_type PURGE';
EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE;
END IF; END;
/
BEGIN EXECUTE IMMEDIATE 'DROP TABLE unit PURGE'; EXCEPTION
WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;
/
BEGIN EXECUTE IMMEDIATE 'DROP TABLE part_category PURGE';
EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE;
END IF; END;
/
BEGIN EXECUTE IMMEDIATE 'DROP TABLE defect PURGE'; EXCEPTION
WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;
/
BEGIN EXECUTE IMMEDIATE 'DROP TABLE work PURGE'; EXCEPTION
WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;
/


-- === Конец безопасного дропа ===
CREATE TABLE brand (
```

```
    id_brand NUMBER PRIMARY KEY,
    name VARCHAR2(50) NOT NULL
);

CREATE TABLE model (
    id_model NUMBER PRIMARY KEY,
    name VARCHAR2(50) NOT NULL,
    id_brand NUMBER NOT NULL,
    CONSTRAINT fk_model_brand FOREIGN KEY (id_brand) REFERENCES
brand(id_brand)
);

CREATE TABLE category (
    id_category NUMBER PRIMARY KEY,
    name VARCHAR2(50) NOT NULL
);

CREATE TABLE body_type (
    id_body_type NUMBER PRIMARY KEY,
    name VARCHAR2(50) NOT NULL
);

CREATE TABLE unit (
    id_unit NUMBER PRIMARY KEY,
    name VARCHAR2(50) NOT NULL,
    abbreviation VARCHAR2(10) NOT NULL
);

CREATE TABLE part_category (
```

```
    id_part_category NUMBER PRIMARY KEY,
    name VARCHAR2(50) NOT NULL
);

-- Владельцы
CREATE TABLE owner (
    id_owner NUMBER PRIMARY KEY,
    surname VARCHAR2(50) NOT NULL,
    name VARCHAR2(50) NOT NULL,
    patronymic VARCHAR2(50),
    phone VARCHAR2(20) NOT NULL
);

-- Автомобили
CREATE TABLE car (
    id_car NUMBER PRIMARY KEY,
    license_plate VARCHAR2(15) UNIQUE NOT NULL,
    year NUMBER(4) NOT NULL,
    id_brand NUMBER NOT NULL,
    id_model NUMBER NOT NULL,
    id_category NUMBER NOT NULL,
    id_body_type NUMBER NOT NULL,
    id_owner NUMBER NOT NULL,
    CONSTRAINT fk_car_brand FOREIGN KEY (id_brand) REFERENCES
brand(id_brand),
    CONSTRAINT fk_car_model FOREIGN KEY (id_model) REFERENCES
model(id_model),
    CONSTRAINT fk_car_category FOREIGN KEY (id_category) REFERENCES
category(id_category),
```

```
    CONSTRAINT fk_car_body_type FOREIGN KEY (id_body_type)
REFERENCES body_type(id_body_type),
    CONSTRAINT fk_car_owner FOREIGN KEY (id_owner) REFERENCES
owner(id_owner)
);

-- Мастера
CREATE TABLE master (
    id_master NUMBER PRIMARY KEY,
    surname VARCHAR2(50) NOT NULL,
    name VARCHAR2(50) NOT NULL,
    patronymic VARCHAR2(50),
    phone VARCHAR2(20),
    birth_date DATE NOT NULL,
    specialization VARCHAR2(100),
    experience_years NUMBER NOT NULL,
    rate_share NUMBER(3,2) NOT NULL -- доля ставки (0.00 - 1.00)
);

-- Запчасти
CREATE TABLE part (
    id_part NUMBER PRIMARY KEY,
    code VARCHAR2(50) UNIQUE NOT NULL,
    name VARCHAR2(100) NOT NULL,
    id_part_category NUMBER NOT NULL,
    characteristics VARCHAR2(255),
    quantity_in_stock NUMBER DEFAULT 0,
    id_unit NUMBER NOT NULL,
    price NUMBER(10,2) NOT NULL, -- цена за единицу
```

```
    CONSTRAINT fk_part_category FOREIGN KEY (id_part_category)
REFERENCES part_category(id_part_category),
    CONSTRAINT fk_part_unit FOREIGN KEY (id_unit) REFERENCES
unit(id_unit)
);

-- Поставщики
CREATE TABLE supplier (
    id_supplier NUMBER PRIMARY KEY,
    name VARCHAR2(100) NOT NULL,
    address VARCHAR2(255) NOT NULL,
    phone VARCHAR2(50),
    manager_surname VARCHAR2(50),
    manager_name VARCHAR2(50),
    manager_patronymic VARCHAR2(50)
);

-- Заказы запчастей у поставщиков
CREATE TABLE supply_order (
    id_supply_order NUMBER PRIMARY KEY,
    id_supplier NUMBER NOT NULL,
    order_date DATE NOT NULL,
    planned_arrival DATE,
    actual_arrival DATE,
    CONSTRAINT fk_supply_supplier FOREIGN KEY (id_supplier)
REFERENCES supplier(id_supplier)
);

CREATE TABLE supply_order_item (
```

```
    id_supply_order NUMBER NOT NULL,
    id_part NUMBER NOT NULL,
    quantity NUMBER NOT NULL,
    cost_in_order NUMBER(10,2) NOT NULL, -- общая стоимость позиции
    PRIMARY KEY (id_supply_order, id_part),
    CONSTRAINT fk_supply_order FOREIGN KEY (id_supply_order)
REFERENCES supply_order(id_supply_order),
    CONSTRAINT fk_supply_part FOREIGN KEY (id_part) REFERENCES
part(id_part)
);


-- Заказы на ремонт
CREATE TABLE repair_order (
    id_repair_order NUMBER PRIMARY KEY,
    id_car NUMBER NOT NULL,
    id_master NUMBER NOT NULL,
    order_date DATE NOT NULL,
    planned_completion DATE NOT NULL,
    actual_completion DATE,
    total_cost NUMBER(12,2) NOT NULL,
    comments VARCHAR2(500),
    CONSTRAINT fk_repair_car FOREIGN KEY (id_car) REFERENCES
car(id_car),
    CONSTRAINT fk_repair_master FOREIGN KEY (id_master) REFERENCES
master(id_master)
);


-- Неисправности (можно хранить как текст, но для гибкости — отдельная
таблица)
CREATE TABLE defect (
```

```
    id_defect NUMBER PRIMARY KEY,
    description VARCHAR2(255) NOT NULL
);

CREATE TABLE repair_defect (
    id_repair_order NUMBER NOT NULL,
    id_defect NUMBER NOT NULL,
    PRIMARY KEY (id_repair_order, id_defect),
    CONSTRAINT fk_rd_order FOREIGN KEY (id_repair_order) REFERENCES
repair_order(id_repair_order),
    CONSTRAINT fk_rd_defect FOREIGN KEY (id_defect) REFERENCES
defect(id_defect)
);

-- Работы
CREATE TABLE work (
    id_work NUMBER PRIMARY KEY,
    name VARCHAR2(100) NOT NULL
);

CREATE TABLE repair_work (
    id_repair_order NUMBER NOT NULL,
    id_work NUMBER NOT NULL,
    PRIMARY KEY (id_repair_order, id_work),
    CONSTRAINT fk_rw_order FOREIGN KEY (id_repair_order) REFERENCES
repair_order(id_repair_order),
    CONSTRAINT fk_rw_work FOREIGN KEY (id_work) REFERENCES
work(id_work)
);
```

```
-- Используемые запчасти в ремонте
CREATE TABLE repair_part (
    id_repair_order NUMBER NOT NULL,
    id_part NUMBER NOT NULL,
    quantity NUMBER NOT NULL,
    PRIMARY KEY (id_repair_order, id_part),
    CONSTRAINT fk_rp_order FOREIGN KEY (id_repair_order) REFERENCES repair_order(id_repair_order),
    CONSTRAINT fk_rp_part FOREIGN KEY (id_part) REFERENCES part(id_part)
);
```

2. Заполнение БД

```
--
=============================================================================
-- 1. СПРАВОЧНИКИ
--
=============================================================================

-- Единицы измерения запчастей: id, полное название, сокращение
INSERT INTO unit VALUES (1, 'штука', 'шт');
INSERT INTO unit VALUES (2, 'комплект', 'компл');

-- Категории запчастей (для анализа и отчётов)
INSERT INTO part_category VALUES (1, 'Тормозная система');
INSERT INTO part_category VALUES (2, 'Подвеска');
INSERT INTO part_category VALUES (3, 'Электрика');
```

```
-- Категории автомобилей (легковой, внедорожник и т.д.)
INSERT INTO category VALUES (1, 'Легковой');
INSERT INTO category VALUES (2, 'Внедорожник');
INSERT INTO category VALUES (3, 'Коммерческий');

-- Типы кузовов
INSERT INTO body_type VALUES (1, 'Седан');
INSERT INTO body_type VALUES (2, 'Хэтчбек');
INSERT INTO body_type VALUES (3, 'Универсал');

-- Марки автомобилей
INSERT INTO brand VALUES (1, 'Toyota');
INSERT INTO brand VALUES (2, 'Honda');
INSERT INTO brand VALUES (3, 'Ford');
INSERT INTO brand VALUES (4, 'BMW');

-- Модели автомобилей (id, название, id_марки)
INSERT INTO model VALUES (1, 'Camry', 1);      -- Toyota
INSERT INTO model VALUES (2, 'Corolla', 1);   -- Toyota
INSERT INTO model VALUES (3, 'Civic', 2);     -- Honda
INSERT INTO model VALUES (4, 'Focus', 3);     -- Ford
INSERT INTO model VALUES (5, 'X5', 4);        -- BMW

--
================================================================
=======
-- 2. ОСНОВНЫЕ СУЩНОСТИ
```

```
--
=============================================================
=======

-- Владельцы: (id, фамилия, имя, отчество, телефон)
INSERT INTO owner VALUES (1, 'Иванов', 'Иван', 'Иванович',
'+79001112233');
INSERT INTO owner VALUES (2, 'Петров', 'Пётр', 'Петрович',
'+79002223344');
INSERT INTO owner VALUES (3, 'Сидоров', 'Сергей', 'Александрович',
'+79003334455');
INSERT INTO owner VALUES (4, 'Кузнецов', 'Андрей', 'Владимирович',
'+79004445566');
INSERT INTO owner VALUES (5, 'Морозов', 'Дмитрий', 'Сергеевич',
'+79005556677');
INSERT INTO owner VALUES (6, 'Соколов', 'Артём', 'Юрьевич',
'+79006667788');
INSERT INTO owner VALUES (7, 'Лебедев', 'Михаил', 'Павлович',
'+79007778899');

-- Автомобили: (id, госномер, год, id_марки, id_модели, id_категории,
id_кузова, id_владельца)
INSERT INTO car VALUES (1, 'А123АА77', 2010, 1, 1, 1, 1, 1); -- Toyota
Camry, Иванов И.И.
INSERT INTO car VALUES (2, 'Б456ББ77', 2015, 1, 2, 1, 1, 2); -- Toyota
Corolla, Петров П.П.
INSERT INTO car VALUES (3, 'В789ВВ77', 2005, 2, 3, 1, 1, 3); -- Honda Civic,
Сидоров С.А.
INSERT INTO car VALUES (4, 'Г012ГГ77', 2020, 3, 4, 1, 1, 4); -- Ford Focus,
Кузнецов А.В.
INSERT INTO car VALUES (5, 'Д345ДД77', 2018, 4, 5, 2, 1, 5); -- BMW X5
(Внедорожник), Морозов Д.С.
```

INSERT INTO car VALUES (6, 'E678EE77', 2018, 1, 1, 1, 1, 1); -- Toyota Camry, Иванов И.И. (второй авто)

INSERT INTO car VALUES (7, 'A124AA77', 2010, 1, 1, 1, 1, 1); -- Toyota Camry, Иванов И.И. (третий авто)

INSERT INTO car VALUES (8, 'Ж111ЖЖ77', 2016, 1, 2, 1, 1, 6); -- Toyota Corolla, Соколов А.Ю.

INSERT INTO car VALUES (9, 'К222КК77', 2019, 3, 4, 1, 1, 7); -- Ford Focus, Лебедев М.П.

INSERT INTO car VALUES (10, 'Л333ЛЛ77', 2021, 3, 5, 2, 1, 7); -- Ford X5, Лебедев М.П.

INSERT INTO car VALUES (11, 'M444MM77', 2015, 1, 2, 1, 1, 6); -- Toyota Corolla 2015, Соколов А.Ю.

-- Мастера: (id, ФИО, телефон, дата_рождения, специализация, стаж, доля_ставки)

INSERT INTO master VALUES (1, 'Смирнов', 'Алексей', 'Олегович', '+79101112233', DATE '1985-03-12', 'Двигатель', 12, 1.0);

INSERT INTO master VALUES (2, 'Козлов', 'Дмитрий', 'Игоревич', '+79102223344', DATE '1990-07-22', 'Ходовая', 8, 0.8);

INSERT INTO master VALUES (3, 'Волков', 'Роман', 'Сергеевич', '+79103334455', DATE '1980-11-05', 'Электрика', 18, 1.0);

INSERT INTO master VALUES (4, 'Орлов', 'Максим', 'Викторович', '+79104445566', DATE '1995-01-30', 'Кузов', 5, 0.5);

-- Неисправности: (id, описание)

INSERT INTO defect VALUES (1, 'Скрип тормозов');

INSERT INTO defect VALUES (2, 'Стук подвески');

INSERT INTO defect VALUES (3, 'Не заводится');

INSERT INTO defect VALUES (4, 'Течь радиатора');

INSERT INTO defect VALUES (5, 'Потеря заряда');

-- Виды работ: (id, название)

```
INSERT INTO work VALUES (1, 'Замена колодок');

INSERT INTO work VALUES (2, 'Замена амортизатора');

INSERT INTO work VALUES (3, 'Диагностика генератора');

INSERT INTO work VALUES (4, 'Замена радиатора');

INSERT INTO work VALUES (5, 'Проверка АКБ');


-- Запчасти: (id, артикул, название, id_категории, характеристики, остаток, id_единицы, цена_за_единицу)

INSERT INTO part VALUES (1, 'P001', 'Колодки тормозные передние', 1, 'Для Camry 2010+', 10, 1, 2500.00);

INSERT INTO part VALUES (2, 'P002', 'Амортизатор передний', 2, 'Для Focus 2020', 5, 1, 4500.00);

INSERT INTO part VALUES (3, 'P003', 'Генератор', 3, 'Универсальный', 3, 1, 12000.00);

INSERT INTO part VALUES (4, 'P004', 'Колодки тормозные задние', 1, 'Для Corolla', 8, 1, 2200.00);

INSERT INTO part VALUES (5, 'P005', 'Радиатор', 3, 'Для X5', 2, 1, 18000.00);

INSERT INTO part VALUES (6, 'P006', 'Лампа H7', 3, 'Для всех авто', 20, 1, 200.00);

INSERT INTO part VALUES (7, 'P007', 'Турбина', 3, 'Для X5', 1, 1, 250000.00);


-- Поставщики: (id, название, адрес, телефон, ФИО_менеджера)

INSERT INTO supplier VALUES (1, 'Авто-Деталь', 'Москва, ул. Ленина 10', '+74951112233', 'Фёдоров', 'Антон', 'Валерьевич');

INSERT INTO supplier VALUES (2, 'Запчасть-Сервис', 'СПб, Невский пр. 50', '+78122223344', 'Григорьев', 'Илья', 'Сергеевич');


--
================================================================
=======

-- 3. ЗАКАЗЫ ПОСТАВОК
```

```
--
================================================================
=======


-- Заказы поставок: (id, id_поставщика, дата_заказа,
планируемая_дата_прибытия, фактическая_дата_прибытия)

-- Заказ 1: от "Авто-Деталь", сделан 10.01.2024, прибыл на 1 день позже срока

INSERT INTO supply_order VALUES (1, 1, DATE '2024-01-10', DATE '2024-
01-15', DATE '2024-01-16');

-- Заказ 2: от "Запчасть-Сервис", сделан 01.02.2024, прибыл на 1 день раньше
срока

INSERT INTO supply_order VALUES (2, 2, DATE '2024-02-01', DATE '2024-
02-05', DATE '2024-02-04');


-- Состав заказов поставок: (id_заказа_поставки, id_запчасти, количество,
общая_стоимость_позиции)

-- Заказ 1 (Авто-Деталь):

--   - 5 шт колодок передних (5 × 2500 = 12 500)

--   - 2 шт генераторов (2 × 12 000 = 24 000)

INSERT INTO supply_order_item VALUES (1, 1, 5, 12500.00);

INSERT INTO supply_order_item VALUES (1, 3, 2, 24000.00);

-- Заказ 2 (Запчасть-Сервис):

--   - 3 шт амортизаторов (3 × 4500 = 13 500)

--   - 1 шт радиатора (1 × 18 000 = 18 000)

INSERT INTO supply_order_item VALUES (2, 2, 3, 13500.00);

INSERT INTO supply_order_item VALUES (2, 5, 1, 18000.00);


--
================================================================
=======

-- 4. ЗАКАЗЫ НА РЕМОНТ
```

```
--
==========================================================================
=======

-- Формат repair_order:

-- (id, id_авто, id_мастера, дата_заказа, планируемая_дата_выполнения,
фактическая_дата_выполнения, итоговая_стоимость, комментарий)


-- АВТОМОБИЛЬ A123AA77 (Toyota Camry, Иванов И.И.) — 3 ремонта, все
с колодками передними

-- Заказ 1: 10.03.2025, план — 12.03, факт — 13.03 → ПРОСРОЧЕН на 1 день

INSERT INTO repair_order VALUES (1, 1, 2, DATE '2025-03-10', DATE '2025-
03-12', DATE '2025-03-13', 3000.00, 'Замена передних колодок');

-- Неисправность: Скрип тормозов (id=1)

INSERT INTO repair_defect VALUES (1, 1);

-- Работа: Замена колодок (id=1)

INSERT INTO repair_work VALUES (1, 1);

-- Запчасть: 1 шт колодок передних (id=1)

INSERT INTO repair_part VALUES (1, 1, 1);


-- Заказ 7: 01.05.2025, план — 02.05, факт — 02.05 → В СРОК → последнее
обращение для A123AA77

INSERT INTO repair_order VALUES (7, 1, 2, DATE '2025-05-01', DATE '2025-
05-02', DATE '2025-05-02', 3000.00, 'Повторная замена');

INSERT INTO repair_defect VALUES (7, 1);

INSERT INTO repair_work VALUES (7, 1);

INSERT INTO repair_part VALUES (7, 1, 1);


-- Заказ 23: 10.03.2025, план — 12.03, факт — 14.03 → ПРОСРОЧЕН на 2 дня
(для статистики по марту)

INSERT INTO repair_order VALUES (23, 1, 1, DATE '2025-03-10', DATE
'2025-03-12', DATE '2025-03-14', 3000.00, 'Просрочено 2');
```

```
INSERT INTO repair_defect VALUES (23, 1);

INSERT INTO repair_work VALUES (23, 1);

INSERT INTO repair_part VALUES (23, 1, 1);


-- АВТОМОБИЛЬ Б456ББ77 (Toyota Corolla, Петров П.П.) — 2 ремонта, оба с колодками задними

-- Заказ 2: 15.03.2025, план — 16.03, факт — 16.03 → В СРОК

INSERT INTO repair_order VALUES (2, 2, 2, DATE '2025-03-15', DATE '2025-03-16', DATE '2025-03-16', 2700.00, 'Замена задних колодок');

INSERT INTO repair_defect VALUES (2, 1);

INSERT INTO repair_work VALUES (2, 1);

INSERT INTO repair_part VALUES (2, 4, 1);


-- Заказ 19: 10.05.2025, план — 11.05, факт — 10.05 → РАНЬШЕ СРОКА → последнее обращение

INSERT INTO repair_order VALUES (19, 2, 2, DATE '2025-05-10', DATE '2025-05-11', DATE '2025-05-10', 2200.00, 'Повторная замена задних колодок');

INSERT INTO repair_defect VALUES (19, 1);

INSERT INTO repair_work VALUES (19, 1);

INSERT INTO repair_part VALUES (19, 4, 1);


-- Заказ 27: 20.05.2025, план — 22.05, факт — 22.05 → В СРОК (для статистики по маю)

INSERT INTO repair_order VALUES (27, 2, 1, DATE '2025-05-20', DATE '2025-05-22', DATE '2025-05-22', 2700.00, 'В срок (May)');

INSERT INTO repair_defect VALUES (27, 1);

INSERT INTO repair_work VALUES (27, 1);

INSERT INTO repair_part VALUES (27, 4, 1);


-- АВТОМОБИЛЬ В789ВВ77 (Honda Civic, Сидоров С.А.) — 2 ремонта генератора + 1 просроченный + 1 раньше срока
```

-- Заказ 3: 20.02.2025, план — 22.02, факт — 21.02 → РАНЬШЕ СРОКА

INSERT INTO repair_order VALUES (3, 3, 3, DATE '2025-02-20', DATE '2025-02-22', DATE '2025-02-21', 13000.00, 'Замена генератора');

INSERT INTO repair_defect VALUES (3, 3);

INSERT INTO repair_work VALUES (3, 3);

INSERT INTO repair_part VALUES (3, 3, 1);

-- Заказ 17: 01.03.2025, план — 03.03, факт — 01.03 → РАНЬШЕ СРОКА → последнее обращение

INSERT INTO repair_order VALUES (17, 3, 3, DATE '2025-03-01', DATE '2025-03-03', DATE '2025-03-01', 12000.00, 'Повторная замена генератора');

INSERT INTO repair_defect VALUES (17, 3);

INSERT INTO repair_work VALUES (17, 3);

INSERT INTO repair_part VALUES (17, 3, 1);

-- Заказ 15: 01.03.2025, план — 03.03, факт — 05.03 → ПРОСРОЧЕН на 2 дня

INSERT INTO repair_order VALUES (15, 3, 1, DATE '2025-03-01', DATE '2025-03-03', DATE '2025-03-05', 3000.00, 'Просрочено');

INSERT INTO repair_defect VALUES (15, 3);

INSERT INTO repair_work VALUES (15, 3);

INSERT INTO repair_part VALUES (15, 3, 1);

-- Заказ 28: 10.05.2025, план — 12.05, факт — 11.05 → РАНЬШЕ СРОКА

INSERT INTO repair_order VALUES (28, 3, 1, DATE '2025-05-10', DATE '2025-05-12', DATE '2025-05-11', 13000.00, 'Раньше срока (Мау)');

INSERT INTO repair_defect VALUES (28, 3);

INSERT INTO repair_work VALUES (28, 3);

INSERT INTO repair_part VALUES (28, 3, 1);

-- АВТОМОБИЛЬ Г012ГГ77 (Ford Focus, Кузнецов А.В.) — 3 ремонта амортизатора

-- Заказ 4: 01.04.2025, план — 03.04, факт — 02.04 → РАНЬШЕ СРОКА

INSERT INTO repair_order VALUES (4, 4, 2, DATE '2025-04-01', DATE '2025-04-03', DATE '2025-04-02', 5000.00, 'Замена переднего амортизатора');

INSERT INTO repair_defect VALUES (4, 2);

INSERT INTO repair_work VALUES (4, 2);

INSERT INTO repair_part VALUES (4, 2, 1);

-- Заказ 18: 01.05.2025, план — 03.05, факт — 02.05 → РАНЬШЕ СРОКА

INSERT INTO repair_order VALUES (18, 4, 2, DATE '2025-05-01', DATE '2025-05-03', DATE '2025-05-02', 4500.00, 'Повторная замена амортизатора');

INSERT INTO repair_defect VALUES (18, 2);

INSERT INTO repair_work VALUES (18, 2);

INSERT INTO repair_part VALUES (18, 2, 1);

-- Заказ 20: 15.05.2025, план — 17.05, факт — 15.05 → РАНЬШЕ СРОКА → последнее обращение

INSERT INTO repair_order VALUES (20, 4, 2, DATE '2025-05-15', DATE '2025-05-17', DATE '2025-05-15', 4500.00, 'Третья замена амортизатора');

INSERT INTO repair_defect VALUES (20, 2);

INSERT INTO repair_work VALUES (20, 2);

INSERT INTO repair_part VALUES (20, 2, 1);

-- Заказ 16: 10.03.2025, план — 12.03, факт — 11.03 → РАНЬШЕ СРОКА (для статистики по марту)

INSERT INTO repair_order VALUES (16, 4, 1, DATE '2025-03-10', DATE '2025-03-12', DATE '2025-03-11', 3000.00, 'Раньше срока');

INSERT INTO repair_defect VALUES (16, 2);

INSERT INTO repair_work VALUES (16, 2);

INSERT INTO repair_part VALUES (16, 2, 1);

-- АВТОМОБИЛЬ Д345ДД77 (BMW X5, Морозов Д.С.) — 3 разные запчасти (лампа, радиатор, турбина)

-- Заказ 10: 20.03.2025, план — 21.03, факт — 20.03 → РАНЬШЕ СРОКА → лампа H7

INSERT INTO repair_order VALUES (10, 5, 3, DATE '2025-03-20', DATE '2025-03-21', DATE '2025-03-20', 300.00, 'Замена лампы на X5');

INSERT INTO repair_defect VALUES (10, 5);

INSERT INTO repair_work VALUES (10, 5);

INSERT INTO repair_part VALUES (10, 6, 1);

-- Заказ 5: 05.04.2025, план — 07.04, факт — 06.04 → РАНЬШЕ СРОКА → радиатор

INSERT INTO repair_order VALUES (5, 5, 3, DATE '2025-04-05', DATE '2025-04-07', DATE '2025-04-06', 19000.00, 'Замена радиатора');

INSERT INTO repair_defect VALUES (5, 4);

INSERT INTO repair_work VALUES (5, 4);

INSERT INTO repair_part VALUES (5, 5, 1);

-- Заказ 9: 05.05.2025, план — 10.05, факт — 05.05 → РАНЬШЕ СРОКА → турбина (самый дорогой заказ)

INSERT INTO repair_order VALUES (9, 5, 3, DATE '2025-05-05', DATE '2025-05-10', DATE '2025-05-05', 260000.00, 'Замена турбины');

INSERT INTO repair_defect VALUES (9, 3);

INSERT INTO repair_work VALUES (9, 3);

INSERT INTO repair_part VALUES (9, 7, 1);

-- Заказ 25: 10.04.2025, план — 12.04, факт — 11.04 → РАНЬШЕ СРОКА (для статистики по апрелю)

INSERT INTO repair_order VALUES (25, 5, 1, DATE '2025-04-10', DATE '2025-04-12', DATE '2025-04-11', 19000.00, 'Раньше срока 2 (Apr)');

```
INSERT INTO repair_defect VALUES (25, 4);

INSERT INTO repair_work VALUES (25, 4);

INSERT INTO repair_part VALUES (25, 5, 1);


-- АВТОМОБИЛЬ Е678ЕЕ77 (Toyota Camry, Иванов И.И.) — 1 ремонт колодок

-- Заказ 21: 10.04.2025, план — 11.04, факт — 10.04 → РАНЬШЕ СРОКА → последнее обращение

INSERT INTO repair_order VALUES (21, 6, 2, DATE '2025-04-10', DATE '2025-04-11', DATE '2025-04-10', 3000.00, 'Замена передних колодок');

INSERT INTO repair_defect VALUES (21, 1);

INSERT INTO repair_work VALUES (21, 1);

INSERT INTO repair_part VALUES (21, 1, 1);


-- Заказ 24: 20.03.2025, план — 22.03, факт — 22.03 → В СРОК (для статистики по марту)

INSERT INTO repair_order VALUES (24, 6, 1, DATE '2025-03-20', DATE '2025-03-22', DATE '2025-03-22', 3000.00, 'В срок 2 (Mar)');

INSERT INTO repair_defect VALUES (24, 1);

INSERT INTO repair_work VALUES (24, 1);

INSERT INTO repair_part VALUES (24, 1, 1);


-- АВТОМОБИЛЬ М444ММ77 (Toyota Corolla 2015, Соколов А.Ю.) — 1 ремонт

-- Заказ 14: 10.01.2025, план — 11.01, факт — 10.01 → РАНЬШЕ СРОКА → последнее обращение

INSERT INTO repair_order VALUES (14, 11, 2, DATE '2025-01-10', DATE '2025-01-11', DATE '2025-01-10', 2700.00, 'Ремонт второй Corolla 2015');

INSERT INTO repair_defect VALUES (14, 1);

INSERT INTO repair_work VALUES (14, 1);

INSERT INTO repair_part VALUES (14, 4, 1);
```

-- Заказ 26: 20.04.2025, план — 22.04, факт — 22.04 → В СРОК (для статистики по апрелю)

INSERT INTO repair_order VALUES (26, 11, 1, DATE '2025-04-20', DATE '2025-04-22', DATE '2025-04-22', 2700.00, 'В срок 2 (Apr)');

INSERT INTO repair_defect VALUES (26, 1);

INSERT INTO repair_work VALUES (26, 1);

INSERT INTO repair_part VALUES (26, 4, 1);

-- Заказ 22: 10.02.2025, план — 12.02, факт — 12.02 → В СРОК (для статистики по февралю)

INSERT INTO repair_order VALUES (22, 2, 2, DATE '2025-02-10', DATE '2025-02-12', DATE '2025-02-12', 2700.00, 'В срок (Feb)');

INSERT INTO repair_defect VALUES (22, 1);

INSERT INTO repair_work VALUES (22, 1);

INSERT INTO repair_part VALUES (22, 4, 1);

-- АВТОМОБИЛЬ Г012ГГ77: дополнительный заказ для мастера Волкова (чтобы он работал с подвеской)

-- Заказ 12: 15.05.2025, план — 17.05, факт — 16.05 → РАНЬШЕ СРОКА, мастер Волков (id=3)

INSERT INTO repair_order VALUES (12, 4, 3, DATE '2025-05-15', DATE '2025-05-17', DATE '2025-05-16', 5000.00, 'Замена амортизатора — Волков');

INSERT INTO repair_defect VALUES (12, 2);

INSERT INTO repair_work VALUES (12, 2);

INSERT INTO repair_part VALUES (12, 2, 1);

Реализация и вызов пакета процедур и функций (PL/SQL Package)

Для реализации логики обработки данных в СУБД Oracle был разработан пакет на языке PL/SQL, объединяющий все процедуры и функции, необходимые для управления данными автосервиса «Автокомплект».

Пакет служит для:

- группировки логически связанных операций в одном модуле;

- инкапсуляции бизнес-логики, скрывающей детали реализации от конечного пользователя;

- повышения безопасности, производительности и удобства сопровождения базы данных.

В рамках проекта «Автокомплект» пакет `autocomplect_queries` реализует следующие функции:

- формирование отчётов по количеству и типам заменённых запчастей для каждого автомобиля;

- вывод перечня запчастей, использованных при ремонте, с указанием даты последнего обращения;

- анализ незадействованных марок и моделей автомобилей в текущем парке;

- выявление автомобилей, на которых многократно заменялась одна и та же запчасть;

- генерация статистики по поставщикам: какие категории запчастей они не поставляют;

- ранжирование мастеров по активности в текущем месяце с учётом ставки;

- определение владельцев, имеющих несколько автомобилей одной модели или одной марки, но разных категорий;

- поиск самых старых автомобилей и запчастей, устанавливаемых на них;

- выявление заказов с максимальной стоимостью, длительностью исполнения или нарушением сроков;

- анализ использования запчастей по маркам, моделям и поставщикам;

- формирование календаря дней без ремонтов в заданном периоде.

Таким образом, пакет обеспечивает полный цикл аналитической и управленческой поддержки работы автосервиса на основе актуальных данных из базы.

```
SET SERVEROUTPUT ON;


CREATE OR REPLACE PACKAGE autocomplect_queries AS
    PROCEDURE query_28;
    PROCEDURE query_29;
    PROCEDURE query_30;
    PROCEDURE query_31;
    PROCEDURE query_32;
    PROCEDURE query_33;
    PROCEDURE query_34;
    PROCEDURE query_35;
    PROCEDURE query_36;
    PROCEDURE query_37;
    PROCEDURE query_38;
    PROCEDURE query_39;
    PROCEDURE query_40;
    PROCEDURE query_41;
    PROCEDURE query_42;
    PROCEDURE query_43;
    PROCEDURE query_44;
    PROCEDURE query_45;
    PROCEDURE query_46;
    PROCEDURE query_47;
    PROCEDURE query_48;
    PROCEDURE query_49;
    PROCEDURE query_50;
```

```
    PROCEDURE query_51;
    PROCEDURE query_52;
END autocomplect_queries;
/

CREATE OR REPLACE PACKAGE BODY autocomplect_queries AS

    -- 28. Госномер, дата последнего обращения, количество заменённых запчастей (если были)
    PROCEDURE query_28 IS
    BEGIN
        DBMS_OUTPUT.PUT_LINE('Задание 28: Выбрать госномер автомобиля, дату последнего обращения и, если были замены запчастей, то их количество.');
        DBMS_OUTPUT.PUT_LINE('Госномер     | Дата обращения | Кол-во запчастей');
        DBMS_OUTPUT.PUT_LINE('-------------|----------------|------------------');
        FOR r IN (
            SELECT
                c.license_plate,
                MAX(ro.order_date) AS last_visit,
                COALESCE(SUM(rp.quantity), 0) AS parts_count
            FROM car c
            LEFT JOIN repair_order ro ON c.id_car = ro.id_car
            LEFT JOIN repair_part rp ON ro.id_repair_order = rp.id_repair_order
            GROUP BY c.license_plate
            ORDER BY c.license_plate
        ) LOOP
            DBMS_OUTPUT.PUT_LINE(
                RPAD(NVL(r.license_plate, 'NULL'), 12) || ' | ' ||
```

```
            TO_CHAR(r.last_visit, 'DD.MM.YY') || '        | ' ||
            LPAD(r.parts_count, 16)
        );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 29. Госномер, дата последнего обращения, наименования запчастей
PROCEDURE query_29 IS
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 29: Выбрать госномер автомобиля, дату последнего обращения и, если были замены запчастей, то наименования запчастей.');
    DBMS_OUTPUT.PUT_LINE('Госномер     | Дата обращения | Запчасти');
DBMS_OUTPUT.PUT_LINE('-------------|----------------|-------------------------------');
    FOR r IN (
        SELECT
            c.license_plate,
            MAX(ro.order_date) AS last_visit,
            LISTAGG(p.name, ', ') WITHIN GROUP (ORDER BY p.name) AS parts_list
        FROM car c
        LEFT JOIN repair_order ro ON c.id_car = ro.id_car
        LEFT JOIN repair_part rp ON ro.id_repair_order = rp.id_repair_order
        LEFT JOIN part p ON rp.id_part = p.id_part
        GROUP BY c.license_plate
        ORDER BY c.license_plate
    ) LOOP
```

```
        DBMS_OUTPUT.PUT_LINE(
            RPAD(NVL(r.license_plate, 'NULL'), 12) || ' | ' ||
            TO_CHAR(r.last_visit, 'DD.MM.YY') || '        | ' ||
            SUBSTR(NVL(r.parts_list, 'нет запчастей'), 1, 40)
        );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 30. Марки и модели, которых нет в БД
PROCEDURE query_30 IS
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 30: Выбрать название марки и название моделей, автомобили которых не представлены в БД.');
    DBMS_OUTPUT.PUT_LINE('Марка        | Модель');
    DBMS_OUTPUT.PUT_LINE('-------------|------------------');
    FOR r IN (
        SELECT b.name AS brand_name, m.name AS model_name
        FROM brand b
        CROSS JOIN model m
        WHERE NOT EXISTS (
            SELECT 1 FROM car c WHERE c.id_brand = b.id_brand AND c.id_model = m.id_model
        )
        ORDER BY b.name, m.name
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(
            RPAD(NVL(r.brand_name, 'NULL'), 12) || ' | ' ||
            r.model_name
```

```
        );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 31. Автомобиль, которому каждый раз меняют одну и ту же запчасть
PROCEDURE query_31 IS
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 31: Выбрать номер автомобиля, которому каждый раз делают замену одной и той же запчасти.');
    DBMS_OUTPUT.PUT_LINE('Госномер');
    DBMS_OUTPUT.PUT_LINE('--------');
    FOR r IN (
        SELECT c.license_plate
        FROM car c
        JOIN repair_order ro ON c.id_car = ro.id_car
        JOIN repair_part rp ON ro.id_repair_order = rp.id_repair_order
        GROUP BY c.id_car, c.license_plate
        HAVING COUNT(DISTINCT rp.id_part) = 1 AND COUNT(*) > 1
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(r.license_plate);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 32. Для каждого поставщика — категории, которые он НЕ поставляет
PROCEDURE query_32 IS
BEGIN
```

```
    DBMS_OUTPUT.PUT_LINE('Задание 32: Для каждого поставщика вывести названия категорий запчастей, которые он не поставляет.');
    DBMS_OUTPUT.PUT_LINE('Поставщик               | Категория');
    DBMS_OUTPUT.PUT_LINE('------------------------|------------------');
    FOR r IN (
      SELECT s.name AS supplier, pc.name AS missing_category
      FROM supplier s
      CROSS JOIN part_category pc
      WHERE NOT EXISTS (
        SELECT 1
        FROM supply_order so
        JOIN supply_order_item soi ON so.id_supply_order = soi.id_supply_order
        JOIN part p ON soi.id_part = p.id_part
        WHERE so.id_supplier = s.id_supplier AND p.id_part_category = pc.id_part_category
      )
      ORDER BY s.name, pc.name
    ) LOOP
      DBMS_OUTPUT.PUT_LINE(
        RPAD(NVL(r.supplier, 'NULL'), 23) || ' | ' ||
        r.missing_category
      );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
  END;

  -- 33. Мастера: сначала без заказов в текущем месяце, потом макс заказов, потом остальные
  PROCEDURE query_33 IS
```

```
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 33: Выбрать фамилии, имена, отчества мастеров и размер ставки. Результат отсортировать следующим образом: в первую очередь тех, кто не выполнил ни одного заказа за текущий месяц, затем тех, кто выполнил максимальное количество заказов за текущий месяц, в последнюю очередь все остальные.');
    DBMS_OUTPUT.PUT_LINE('ФИО мастера                    | Ставка');
    DBMS_OUTPUT.PUT_LINE('-------------------------------|--------');
    FOR r IN (
      SELECT m.surname, m.name, m.patronymic, m.rate_share,
          COUNT(ro.id_repair_order) AS orders_count
      FROM master m
      LEFT JOIN repair_order ro ON m.id_master = ro.id_master
        AND EXTRACT(YEAR FROM ro.order_date) = EXTRACT(YEAR FROM SYSDATE)
        AND EXTRACT(MONTH FROM ro.order_date) = EXTRACT(MONTH FROM SYSDATE)
      GROUP BY m.id_master, m.surname, m.name, m.patronymic, m.rate_share
      ORDER BY
        CASE WHEN COUNT(ro.id_repair_order) = 0 THEN 0
          WHEN COUNT(ro.id_repair_order) = (
            SELECT MAX(cnt) FROM (
              SELECT COUNT(*) AS cnt
              FROM repair_order
              WHERE EXTRACT(YEAR FROM order_date) = EXTRACT(YEAR FROM SYSDATE)
                AND EXTRACT(MONTH FROM order_date) = EXTRACT(MONTH FROM SYSDATE)
              GROUP BY id_master
            )
```

```
            ) THEN 1
            ELSE 2
        END,
        orders_count DESC
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(
            RPAD(NVL(r.surname || ' ' || r.name || ' ' || NVL(r.patronymic, ''), 'NULL'), 31) || ' | ' ||
            LPAD(TO_CHAR(r.rate_share, '990.00'), 6)
        );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 34. Для каждого поставщика — все категории + кол-во запчастей (0 если нет)
PROCEDURE query_34 IS
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 34: Для каждого поставщика выбрать названия всех категорий запчастей и количество различных запчастей...');
    DBMS_OUTPUT.PUT_LINE('Поставщик                | Категория        | Кол-во');
    DBMS_OUTPUT.PUT_LINE('-----------------------|------------------|-------');
    FOR r IN (
        SELECT
            s.name AS supplier,
            pc.name AS category,
            COALESCE(COUNT(DISTINCT p.id_part), 0) AS part_count
        FROM supplier s
```

```
        CROSS JOIN part_category pc
        LEFT JOIN supply_order so ON s.id_supplier = so.id_supplier
        LEFT JOIN supply_order_item soi ON so.id_supply_order =
soi.id_supply_order
        LEFT JOIN part p ON soi.id_part = p.id_part AND p.id_part_category =
pc.id_part_category
        GROUP BY s.id_supplier, s.name, pc.id_part_category, pc.name
        ORDER BY s.id_supplier, pc.name
      ) LOOP
        DBMS_OUTPUT.PUT_LINE(
          RPAD(NVL(r.supplier, 'NULL'), 23) || ' | ' ||
          RPAD(NVL(r.category, 'NULL'), 16) || ' | ' ||
          LPAD(r.part_count, 5)
        );
      END LOOP;
      DBMS_OUTPUT.PUT_LINE(' ');
    END;

    -- 35. Владельцы с двумя авто одной модели
    PROCEDURE query_35 IS
    BEGIN
      DBMS_OUTPUT.PUT_LINE('Задание 35: Выбрать фамилию, имя,
отчество владельца двух автомобилей одной модели.');
      DBMS_OUTPUT.PUT_LINE('ФИО владельца');
      DBMS_OUTPUT.PUT_LINE('----------------------------');
      FOR r IN (
        SELECT o.surname, o.name, o.patronymic
        FROM owner o
        JOIN car c ON o.id_owner = c.id_owner
```

```
        GROUP BY o.id_owner, o.surname, o.name, o.patronymic
        HAVING COUNT(*) >= 2
          AND COUNT(DISTINCT c.id_model) = 1
      ) LOOP
        DBMS_OUTPUT.PUT_LINE(r.surname || ' ' || r.name || ' ' ||
NVL(r.patronymic, ''));
      END LOOP;
      DBMS_OUTPUT.PUT_LINE(' ');
    END;

    -- 36. Авто одной модели и года, отремонтированные одинаковое число раз
    PROCEDURE query_36 IS
    BEGIN
      DBMS_OUTPUT.PUT_LINE('Задание 36: Выбрать госномера, названия
модели и марки автомобилей одной модели и одного года выпуска, которые
ремонтировались одинаковое количество раз.');
      DBMS_OUTPUT.PUT_LINE('Госномер     | Марка   | Модель');
      DBMS_OUTPUT.PUT_LINE('-------------|---------|--------');
      FOR r IN (
        SELECT c.license_plate, b.name AS brand, m.name AS model
        FROM car c
        JOIN model m ON c.id_model = m.id_model
        JOIN brand b ON c.id_brand = b.id_brand
        JOIN (
          SELECT id_car, COUNT(*) AS cnt
          FROM repair_order
          GROUP BY id_car
        ) ro ON c.id_car = ro.id_car
        WHERE EXISTS (
```

```
            SELECT 1
            FROM car c2
            JOIN (
               SELECT id_car, COUNT(*) AS cnt
               FROM repair_order
               GROUP BY id_car
            ) ro2 ON c2.id_car = ro2.id_car
            WHERE c2.id_model = c.id_model
              AND c2.year = c.year
              AND ro2.cnt = ro.cnt
              AND c2.id_car != c.id_car
         )
         ORDER BY b.name, m.name, c.license_plate
      ) LOOP
         DBMS_OUTPUT.PUT_LINE(
            RPAD(r.license_plate, 12) || ' | ' ||
            RPAD(r.brand, 7) || ' | ' ||
            r.model
         );
      END LOOP;
      DBMS_OUTPUT.PUT_LINE(' ');
   END;

   -- 37. Самая часто ремонтируемая марка+модель
   PROCEDURE query_37 IS
   BEGIN
      DBMS_OUTPUT.PUT_LINE('Задание 37: Выбрать название марки и модели, автомобили которой ремонтировались чаще других.');
      DBMS_OUTPUT.PUT_LINE('Марка   | Модель');
```

```
        DBMS_OUTPUT.PUT_LINE('--------|--------');
        FOR r IN (
            SELECT b.name AS brand, m.name AS model
            FROM car c
            JOIN brand b ON c.id_brand = b.id_brand
            JOIN model m ON c.id_model = m.id_model
            JOIN repair_order ro ON c.id_car = ro.id_car
            GROUP BY b.name, m.name
            ORDER BY COUNT(*) DESC
            FETCH FIRST 1 ROW ONLY
        ) LOOP
            DBMS_OUTPUT.PUT_LINE(RPAD(r.brand, 7) || ' | ' || r.model);
        END LOOP;
        DBMS_OUTPUT.PUT_LINE(' ');
    END;

    -- 38. Авто без замены запчастей
    PROCEDURE query_38 IS
    BEGIN
        DBMS_OUTPUT.PUT_LINE('Задание 38: Выбрать все данные об
автомобилях, для которых не производились замены запчастей.');
        DBMS_OUTPUT.PUT_LINE('Госномер     | Год | Марка   | Модель  |
Владелец');
        DBMS_OUTPUT.PUT_LINE('-------------|-----|---------|---------|----------');
        FOR r IN (
            SELECT c.license_plate, c.year, b.name AS brand, m.name AS model,
o.surname
            FROM car c
            JOIN brand b ON c.id_brand = b.id_brand
```

```
        JOIN model m ON c.id_model = m.id_model
        JOIN owner o ON c.id_owner = o.id_owner
        LEFT JOIN repair_order ro ON c.id_car = ro.id_car
        LEFT JOIN repair_part rp ON ro.id_repair_order = rp.id_repair_order
        WHERE rp.id_repair_order IS NULL
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(
            RPAD(r.license_plate, 12) || ' | ' ||
            LPAD(r.year, 3) || ' | ' ||
            RPAD(r.brand, 7) || ' | ' ||
            RPAD(r.model, 7) || ' | ' ||
            r.surname
        );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 39. Заказ с самым долгим сроком исполнения
PROCEDURE query_39 IS
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 39: Выбрать все данные о заказе с самым продолжительным сроком исполнения.');
    DBMS_OUTPUT.PUT_LINE('ID заказа | Авто       | Длительность (дн)');
    DBMS_OUTPUT.PUT_LINE('----------|------------|------------------');
    FOR r IN (
        SELECT ro.id_repair_order, c.license_plate, (ro.actual_completion - ro.order_date) AS duration
        FROM repair_order ro
        JOIN car c ON ro.id_car = c.id_car
```

```
        WHERE ro.actual_completion IS NOT NULL
        ORDER BY (ro.actual_completion - ro.order_date) DESC
        FETCH FIRST 1 ROW ONLY
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(
            LPAD(r.id_repair_order, 9) || ' | ' ||
            RPAD(r.license_plate, 10) || ' | ' ||
            LPAD(r.duration, 18)
        );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 40. Самый дорогой заказ — владелец и авто
PROCEDURE query_40 IS
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 40: Выбрать фамилию, имя, отчество владельца, госномер автомобиля, на который был сделан самый дорогой заказ.');
    DBMS_OUTPUT.PUT_LINE('ФИО владельца              | Госномер');
    DBMS_OUTPUT.PUT_LINE('---------------------------|----------');
    FOR r IN (
        SELECT o.surname, o.name, o.patronymic, c.license_plate
        FROM repair_order ro
        JOIN car c ON ro.id_car = c.id_car
        JOIN owner o ON c.id_owner = o.id_owner
        ORDER BY ro.total_cost DESC
        FETCH FIRST 1 ROW ONLY
    ) LOOP
```

```
        DBMS_OUTPUT.PUT_LINE(
            RPAD(r.surname || ' ' || r.name || ' ' || NVL(r.patronymic, ''), 27) || ' | ' ||
            r.license_plate
        );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 41. Самая часто меняющаяся запчасть для модели
PROCEDURE query_41 IS
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 41: Выбрать название марки, название модели и название запчасти, которую приходится менять для данной модели чаще других.');
    DBMS_OUTPUT.PUT_LINE('Марка   | Модель  | Запчасть');
    DBMS_OUTPUT.PUT_LINE('--------|---------|----------------------');
    FOR r IN (
        SELECT b.name AS brand, m.name AS model, p.name AS part
        FROM repair_part rp
        JOIN repair_order ro ON rp.id_repair_order = ro.id_repair_order
        JOIN car c ON ro.id_car = c.id_car
        JOIN brand b ON c.id_brand = b.id_brand
        JOIN model m ON c.id_model = m.id_model
        JOIN part p ON rp.id_part = p.id_part
        GROUP BY b.name, m.name, p.name
        ORDER BY COUNT(*) DESC
        FETCH FIRST 1 ROW ONLY
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(
```

```
            RPAD(r.brand, 7) || ' | ' ||
            RPAD(r.model, 7) || ' | ' ||
            SUBSTR(r.part, 1, 22)
        );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 42. За последний год: просроченные / в срок / раньше срока
PROCEDURE query_42 IS
    v_year NUMBER := EXTRACT(YEAR FROM SYSDATE);
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 42: Для последнего года выбрать количество просроченных заказов, в срок, раньше срока.');
    DBMS_OUTPUT.PUT_LINE('Тип заказа               | Кол-во');
    DBMS_OUTPUT.PUT_LINE('-------------------------|-------');
    FOR r IN (
        SELECT 'Просроченные заказы' AS type, COUNT(*) AS cnt
        FROM repair_order
        WHERE EXTRACT(YEAR FROM order_date) = v_year
          AND actual_completion > planned_completion
        UNION ALL
        SELECT 'Заказы в срок', COUNT(*)
        FROM repair_order
        WHERE EXTRACT(YEAR FROM order_date) = v_year
          AND actual_completion = planned_completion
        UNION ALL
        SELECT 'Заказы раньше срока', COUNT(*)
        FROM repair_order
```

```
        WHERE EXTRACT(YEAR FROM order_date) = v_year
          AND actual_completion < planned_completion
      ) LOOP
        DBMS_OUTPUT.PUT_LINE(RPAD(r.type, 25) || ' | ' || LPAD(r.cnt, 5));
      END LOOP;
      DBMS_OUTPUT.PUT_LINE(' ');
    END;

    -- 43. По месяцам — статистика выполнения
    PROCEDURE query_43 IS
    BEGIN
      DBMS_OUTPUT.PUT_LINE('Задание 43: Для каждого месяца выбрать количество просроченных заказов, в срок, раньше срока.');
      DBMS_OUTPUT.PUT_LINE('Месяц      | Тип заказа            | Кол-во');
      DBMS_OUTPUT.PUT_LINE('-----------|-----------------------|-------');
      FOR r IN (
        SELECT
          TO_CHAR(order_date, 'Mon') AS month_abbr,
          CASE
            WHEN actual_completion > planned_completion THEN 'Просроченные заказы'
            WHEN actual_completion = planned_completion THEN 'Заказы в срок'
            ELSE 'Заказы раньше срока'
          END AS status,
          COUNT(*) AS cnt
        FROM repair_order
        WHERE actual_completion IS NOT NULL
        GROUP BY TO_CHAR(order_date, 'Mon'),
```

```
            CASE
                WHEN actual_completion > planned_completion THEN 'Просроченные заказы'
                WHEN actual_completion = planned_completion THEN 'Заказы в срок'
                ELSE 'Заказы раньше срока'
            END
        ORDER BY MIN(order_date)
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(
            RPAD(r.month_abbr, 10) || ' | ' ||
            RPAD(r.status, 21) || ' | ' ||
            LPAD(r.cnt, 5)
        );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 44. Авто, на котором ни одну запчасть не меняли дважды
PROCEDURE query_44 IS
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 44: Выбрать автомобиль, на котором ни одну запчасть не меняли дважды.');
    DBMS_OUTPUT.PUT_LINE('Госномер');
    DBMS_OUTPUT.PUT_LINE('--------');
    FOR r IN (
        SELECT c.license_plate
        FROM car c
        JOIN repair_order ro ON c.id_car = ro.id_car
```

```
        JOIN repair_part rp ON ro.id_repair_order = rp.id_repair_order
        GROUP BY c.id_car, c.license_plate
        HAVING MAX(rp.quantity) = 1 AND COUNT(*) = COUNT(DISTINCT
rp.id_part)
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(r.license_plate);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
  END;

  -- 45. Мастер, чинивший авто всех категорий
  PROCEDURE query_45 IS
    v_total_categories NUMBER;
  BEGIN
    SELECT COUNT(*) INTO v_total_categories FROM category;
    DBMS_OUTPUT.PUT_LINE('Задание 45: Выбрать фамилию, имя,
отчество мастера, который чинил автомобили всех категорий.');
    DBMS_OUTPUT.PUT_LINE('ФИО мастера');
    DBMS_OUTPUT.PUT_LINE('----------------------------');
    FOR r IN (
        SELECT m.surname, m.name, m.patronymic
        FROM master m
        JOIN repair_order ro ON m.id_master = ro.id_master
        JOIN car c ON ro.id_car = c.id_car
        GROUP BY m.id_master, m.surname, m.name, m.patronymic
        HAVING COUNT(DISTINCT c.id_category) = v_total_categories
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(r.surname || ' ' || r.name || ' ' ||
NVL(r.patronymic, ''));
```

```
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 46. Владельцы с 2 авто разных категорий, но одной марки
PROCEDURE query_46 IS
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 46: Выбрать фамилии, имена, отчества владельцев транспортных средств двух разных категорий, которые имеют два автомобиля одной марки.');
    DBMS_OUTPUT.PUT_LINE('ФИО владельца');
    DBMS_OUTPUT.PUT_LINE('----------------------------');
    FOR r IN (
        SELECT o.surname, o.name, o.patronymic
        FROM owner o
        JOIN car c ON o.id_owner = c.id_owner
        GROUP BY o.id_owner, o.surname, o.name, o.patronymic
        HAVING COUNT(*) >= 2
          AND COUNT(DISTINCT c.id_category) >= 2
          AND COUNT(DISTINCT c.id_brand) = 1
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(r.surname || ' ' || r.name || ' ' || NVL(r.patronymic, ''));
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 47. Запчасти на самые старые авто
PROCEDURE query_47 IS
```

```
    v_min_year NUMBER;
  BEGIN
    SELECT MIN(year) INTO v_min_year FROM car;
    DBMS_OUTPUT.PUT_LINE('Задание 47: Выбрать названия запчастей, которые устанавливались на самые старые автомобили.');
    DBMS_OUTPUT.PUT_LINE('Запчасть');
    DBMS_OUTPUT.PUT_LINE('--------------------------------------');
    FOR r IN (
      SELECT DISTINCT p.name
      FROM part p
      JOIN repair_part rp ON p.id_part = rp.id_part
      JOIN repair_order ro ON rp.id_repair_order = ro.id_repair_order
      JOIN car c ON ro.id_car = c.id_car
      WHERE c.year = v_min_year
      ORDER BY p.name
    ) LOOP
      DBMS_OUTPUT.PUT_LINE(SUBSTR(r.name, 1, 40));
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
  END;


  -- 48. Авто, на которых меняли и самую дорогую, и самую дешёвую запчасть
  PROCEDURE query_48 IS
    v_max_price NUMBER;
    v_min_price NUMBER;
  BEGIN
    SELECT MAX(price) INTO v_max_price FROM part;
    SELECT MIN(price) INTO v_min_price FROM part;
```

```
    DBMS_OUTPUT.PUT_LINE('Задание 48: Выбрать марку, модель, госномер автомобилей, на которых делали замену, как самой дорогой, так и самой дешевой запчасти.');
    DBMS_OUTPUT.PUT_LINE('Марка   | Модель  | Госномер');
    DBMS_OUTPUT.PUT_LINE('--------|---------|----------');
    FOR r IN (
      SELECT DISTINCT c.license_plate, b.name AS brand, m.name AS model
      FROM car c
      JOIN repair_order ro ON c.id_car = ro.id_car
      JOIN repair_part rp ON ro.id_repair_order = rp.id_repair_order
      JOIN part p ON rp.id_part = p.id_part
      JOIN brand b ON c.id_brand = b.id_brand
      JOIN model m ON c.id_model = m.id_model
      WHERE c.id_car IN (
        SELECT c1.id_car
        FROM car c1
        JOIN repair_order ro1 ON c1.id_car = ro1.id_car
        JOIN repair_part rp1 ON ro1.id_repair_order = rp1.id_repair_order
        JOIN part p1 ON rp1.id_part = p1.id_part
        WHERE p1.price = v_max_price
      )
      AND c.id_car IN (
        SELECT c2.id_car
        FROM car c2
        JOIN repair_order ro2 ON c2.id_car = ro2.id_car
        JOIN repair_part rp2 ON ro2.id_repair_order = rp2.id_repair_order
        JOIN part p2 ON rp2.id_part = p2.id_part
        WHERE p2.price = v_min_price
      )
```

```
    ) LOOP
        DBMS_OUTPUT.PUT_LINE(
            RPAD(r.brand, 7) || ' | ' ||
            RPAD(r.model, 7) || ' | ' ||
            r.license_plate
        );
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
END;

-- 49. Владельцы с ≥2 авто, без тёзок/однофамильцев
PROCEDURE query_49 IS
BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 49: Вывести фамилии, имена, отчества владельцев, которые обладают 2 автомобилями и более, но не имеют тезок и/или однофамильцев.');
    DBMS_OUTPUT.PUT_LINE('ФИО владельца');
    DBMS_OUTPUT.PUT_LINE('----------------------------');
    FOR r IN (
        SELECT o.surname, o.name, o.patronymic
        FROM owner o
        WHERE (SELECT COUNT(*) FROM car WHERE id_owner = o.id_owner) >= 2
          AND NOT EXISTS (
            SELECT 1 FROM owner o2
            WHERE o2.id_owner != o.id_owner
              AND (o2.surname = o.surname OR o2.name = o.name)
          )
    ) LOOP
```

```
        DBMS_OUTPUT.PUT_LINE(r.surname || ' ' || r.name || ' ' ||
NVL(r.patronymic, ''));
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
  END;

  -- 50. Три самых старых авто
  PROCEDURE query_50 IS
  BEGIN
    DBMS_OUTPUT.PUT_LINE('Задание 50: Выбрать тройку самых старых
автомобилей.');
    DBMS_OUTPUT.PUT_LINE('Госномер     | Год выпуска');
    DBMS_OUTPUT.PUT_LINE('-------------|------------');
    FOR r IN (
      SELECT license_plate, year
      FROM car
      ORDER BY year ASC
      FETCH FIRST 3 ROWS ONLY
    ) LOOP
      DBMS_OUTPUT.PUT_LINE(RPAD(r.license_plate, 12) || ' | ' || r.year);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
  END;

  -- 51. Даты прошлого месяца без замены запчастей
  PROCEDURE query_51 IS
    v_year NUMBER := EXTRACT(YEAR FROM
ADD_MONTHS(SYSDATE, -1));
    v_month NUMBER := EXTRACT(MONTH FROM
ADD_MONTHS(SYSDATE, -1));
```

```
    v_days NUMBER;
  BEGIN
    SELECT EXTRACT(DAY FROM
LAST_DAY(ADD_MONTHS(SYSDATE, -1))) INTO v_days FROM dual;
    DBMS_OUTPUT.PUT_LINE('Задание 51: Выбрать все даты прошлого
месяца, в которые не осуществляли замену запчастей.');
    DBMS_OUTPUT.PUT_LINE('Дата');
    DBMS_OUTPUT.PUT_LINE('----------');
    FOR d IN 1..v_days LOOP
      DECLARE
        v_date DATE := TO_DATE(v_year || '-' || LPAD(v_month, 2, '0') || '-' ||
LPAD(d, 2, '0'), 'YYYY-MM-DD');
        v_exists NUMBER;
      BEGIN
        SELECT COUNT(*) INTO v_exists
        FROM repair_part rp
        JOIN repair_order ro ON rp.id_repair_order = ro.id_repair_order
        WHERE ro.order_date = v_date;
        IF v_exists = 0 THEN
          DBMS_OUTPUT.PUT_LINE(TO_CHAR(v_date,
'DD.MM.YYYY'));
        END IF;
      END;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(' ');
  END;

  -- 52. Поставщики: заказы в прошлом году, % от общего
  PROCEDURE query_52 IS
    v_total_orders NUMBER;
```

```
    v_year NUMBER := EXTRACT(YEAR FROM SYSDATE) - 1;
  BEGIN
    SELECT COUNT(*) INTO v_total_orders
    FROM supply_order
    WHERE EXTRACT(YEAR FROM order_date) = v_year;

    DBMS_OUTPUT.PUT_LINE('Задание 52: Выбрать названия
поставщиков, количество заказов в прошлом году, процентное отношение ко
всем заказам прошлого года.');
    DBMS_OUTPUT.PUT_LINE('Поставщик              | Заказов | Процент');
    DBMS_OUTPUT.PUT_LINE('-----------------------|---------|--------');
    FOR r IN (
      SELECT
        s.name,
        COUNT(so.id_supply_order) AS orders,
        ROUND(COUNT(so.id_supply_order) * 100.0 /
NULLIF(v_total_orders, 0), 2) AS percent
      FROM supplier s
      LEFT JOIN supply_order so ON s.id_supplier = so.id_supplier
        AND EXTRACT(YEAR FROM so.order_date) = v_year
      GROUP BY s.id_supplier, s.name
      ORDER BY orders DESC
    ) LOOP
      DBMS_OUTPUT.PUT_LINE(
        RPAD(NVL(r.name, 'NULL'), 23) || ' | ' ||
        LPAD(NVL(r.orders, 0), 7) || ' | ' ||
        LPAD(NVL(r.percent, 0), 6)
      );
    END LOOP;
```

```
        DBMS_OUTPUT.PUT_LINE(' ');
    END;

END autocomplect_queries;
/

BEGIN
    DBMS_OUTPUT.PUT_LINE(' ');
    autocomplect_queries.query_28;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_29;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_30;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_31;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_32;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_33;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_34;
    DBMS_OUTPUT.PUT_LINE(' ');
```

```
autocomplect_queries.query_35;
DBMS_OUTPUT.PUT_LINE(' ');

autocomplect_queries.query_36;
DBMS_OUTPUT.PUT_LINE(' ');

autocomplect_queries.query_37;
DBMS_OUTPUT.PUT_LINE(' ');

autocomplect_queries.query_38;
DBMS_OUTPUT.PUT_LINE(' ');

autocomplect_queries.query_39;
DBMS_OUTPUT.PUT_LINE(' ');

autocomplect_queries.query_40;
DBMS_OUTPUT.PUT_LINE(' ');

autocomplect_queries.query_41;
DBMS_OUTPUT.PUT_LINE(' ');

autocomplect_queries.query_42;
DBMS_OUTPUT.PUT_LINE(' ');

autocomplect_queries.query_43;
DBMS_OUTPUT.PUT_LINE(' ');
```

```
    autocomplect_queries.query_44;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_45;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_46;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_47;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_48;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_49;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_50;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_51;
    DBMS_OUTPUT.PUT_LINE(' ');

    autocomplect_queries.query_52;
    DBMS_OUTPUT.PUT_LINE(' ');
END;
/
```

Демонстрация процедур

Задание 28: Выбрать госномер автомобиля, дату последнего обращения и, если были замены запчастей, то их количество.

Госномер | Дата обращения | Кол-во запчастей

---|---|---

А123АА77 | 01.05.25 | 3

А124АА77 | | 0

Б456ББ77 | 20.05.25 | 4

В789ВВ77 | 10.05.25 | 4

Г012ГГ77 | 15.05.25 | 5

Д345ДД77 | 05.05.25 | 4

Е678ЕЕ77 | 10.04.25 | 2

Ж111ЖЖ77 | | 0

К222КК77 | | 0

Л333ЛЛ77 | | 0

М444ММ77 | 20.04.25 | 3

Задание 29: Выбрать госномер автомобиля, дату последнего обращения и, если были замены запчастей, то наименования запчастей.

Госномер | Дата обращения | Запчасти

---|---|---

А123АА77 | 01.05.25 | Колодки тормозные передние, Колодки торм

А124АА77 | | нет запчастей

Б456ББ77 | 20.05.25 | Колодки тормозные задние, Колодки тормоз

В789ВВ77 | 10.05.25 | Генератор, Генератор, Генератор, Генерат

Г012ГГ77 | 15.05.25 | Амортизатор передний, Амортизатор передн

Д345ДД77 | 05.05.25 | Лампа H7, Радиатор, Радиатор, Турбина

Е678ЕЕ77 | 10.04.25 | Колодки тормозные передние, Колодки торм

Ж111ЖЖ77 | | нет запчастей

К222КК77 | | нет запчастей

Л333ЛЛ77 | | нет запчастей

М444ММ77 | 20.04.25 | Колодки тормозные задние, Колодки тормоз

Задание 30: Выбрать название марки и название моделей, автомобили которых не представлены в БД.

Марка | Модель

---|---

BMW | Camry

BMW | Civic

BMW | Corolla

BMW | Focus

Ford | Camry

Ford | Civic

Ford | Corolla

Honda | Camry

Honda | Corolla

Honda | Focus

Honda | X5

Toyota | Civic

Toyota | Focus

Toyota | X5

Задание 31: Выбрать номер автомобиля, которому каждый раз делают замену одной и той же запчасти.

Госномер

А123АА77

Б456ББ77

В789ВВ77

Г012ГГ77

Е678ЕЕ77

М444ММ77

Задание 32: Для каждого поставщика вывести названия категорий запчастей, которые он не поставляет.

Поставщик | Категория

---|---

Авто-Деталь | Подвеска

Запчасть-Сервис | Тормозная система

Задание 33: Выбрать фамилии, имена, отчества мастеров и размер ставки. Результат отсортировать следующим образом: в первую очередь тех, кто не выполнил ни одного заказа за текущий месяц, затем тех, кто выполнил максимальное количество заказов за текущий месяц, в последнюю очередь все остальные.

ФИО мастера | Ставка

---|---

Смирнов Алексей Олегович | 1.0

Орлов Максим Викторович | 0.5

Волков Роман Сергеевич | 1.0

Козлов Дмитрий Игоревич | 0.8

Задание 34: Для каждого поставщика выбрать названия всех категорий запчастей и количество различных запчастей...

Поставщик | Категория | Кол-во

----------|----------|-------

Авто-Деталь | Подвеска | 0

Авто-Деталь | Тормозная систем | 1

Авто-Деталь | Электрика | 1

Запчасть-Сервис | Подвеска | 1

Запчасть-Сервис | Тормозная систем | 0

Запчасть-Сервис | Электрика | 1

Задание 35: Выбрать фамилию, имя, отчество владельца двух автомобилей одной модели.

ФИО владельца

Иванов Иван Иванович

Соколов Артём Юрьевич

Задание 36: Выбрать госномера, названия модели и марки автомобилей одной модели и одного года выпуска, которые ремонтировались одинаковое количество раз.

Госномер | Марка | Модель

----------|---------|--------

А123АА77 | Toyota | Camry

Б456ББ77 | Toyota | Corolla

Задание 37: Выбрать название марки и модели, автомобили которой ремонтировались чаще других.

Марка | Модель

--------|--------

Toyota | Corolla

Задание 38: Выбрать все данные об автомобилях, для которых не производились замены запчастей.

Госномер | Год | Марка | Модель | Владелец

-------------|-----|---------|---------|----------

А124АА77 | 201 | Toyota | Camry | Иванов

Ж111ЖЖ77 | 201 | Toyota | Corolla | Соколов

К222КК77 | 201 | Ford | Focus | Лебедев

Л333ЛЛ77 | 202 | Ford | X5 | Лебедев

Задание 39: Выбрать все данные о заказе с самым продолжительным сроком исполнения.

ID заказа | Авто | Длительность (дн)

----------|------------|------------------

23 | А123АА77 | 4

Задание 40: Выбрать фамилию, имя, отчество владельца, госномер автомобиля, на который был сделан самый дорогой заказ.

ФИО владельца | Госномер

---------------------------|----------

Морозов Дмитрий Сергеевич | Д345ДД77

Задание 41: Выбрать название марки, название модели и название запчасти, которую приходится менять для данной модели чаще других.

Марка | Модель | Запчасть

--------|---------|----------------------

Toyota | Corolla | Колодки тормозные задн

Задание 42: Для последнего года выбрать количество просроченных заказов, в срок, раньше срока.

Тип заказа | Кол-во

-------------------------|-------

Просроченные заказы | 3

Заказы в срок | 7

Заказы раньше срока | 15

Задание 43: Для каждого месяца выбрать количество просроченных заказов, в срок, раньше срока.

Месяц | Тип заказа | Кол-во

-----------|----------------------|-------

Jan | Заказы раньше срока | 1

Feb | Заказы в срок | 2

Feb | Заказы раньше срока | 1

Mar | Заказы раньше срока | 3

Mar | Просроченные заказы | 3

Mar | Заказы в срок | 2

Apr | Заказы раньше срока | 4

Apr | Заказы в срок | 1

May | Заказы раньше срока | 6

May | Заказы в срок | 2

Задание 44: Выбрать автомобиль, на котором ни одну запчасть не меняли дважды.

Госномер

Б456ББ77

Задание 45: Выбрать фамилию, имя, отчество мастера, который чинил автомобили всех категорий.

ФИО мастера

Смирнов Алексей Олегович

Задание 46: Выбрать фамилии, имена, отчества владельцев транспортных средств двух разных категорий, которые имеют два автомобиля одной марки.

ФИО владельца

Лебедев Михаил Павлович

Задание 47: Выбрать названия запчастей, которые устанавливались на самые старые автомобили.

Запчасть

Генератор

Задание 48: Выбрать марку, модель, госномер автомобилей, на которых делали замену, как самой дорогой, так и самой дешевой запчасти.

Марка | Модель | Госномер

--------|---------|----------

BMW | X5 | Д345ДД77

Задание 49: Вывести фамилии, имена, отчества владельцев, которые обладают 2 автомобилями и более, но не имеют тезок и/или однофамильцев.

ФИО владельца

Иванов Иван Иванович

Соколов Артём Юрьевич

Лебедев Михаил Павлович

Задание 50: Выбрать тройку самых старых автомобилей.

Госномер | Год выпуска

----------|----------

B789BB77 | 2005

A123AA77 | 2010

A124AA77 | 2010

Задание 51: Выбрать все даты прошлого месяца, в которые не осуществляли замену запчастей.

Дата

01.10.2025

02.10.2025

03.10.2025

04.10.2025

05.10.2025

06.10.2025

07.10.2025

08.10.2025

09.10.2025

10.10.2025

11.10.2025

12.10.2025

13.10.2025

14.10.2025

15.10.2025

16.10.2025

17.10.2025

18.10.2025

19.10.2025

20.10.2025

21.10.2025

22.10.2025

23.10.2025

24.10.2025

25.10.2025

26.10.2025

27.10.2025

28.10.2025

29.10.2025

30.10.2025

31.10.2025

Задание 52: Выбрать названия поставщиков, количество заказов в прошлом году, процентное отношение ко всем заказам прошлого года.

Поставщик | Заказов | Процент

----------|---------|--------

Запчасть-Сервис | 1 | 50

Авто-Деталь | 1 | 50

Вывод

В ходе работы спроектирована и реализована реляционная база данных автосервиса «Автокомплект» в СУБД Oracle. Построена ER-модель, определены основные сущности (владельцы, автомобили, марки, модели, категории, типы кузовов, мастера, запчасти, категории запчастей, поставщики, заказы на поставку и на ремонт) и связи между ними. Заданы первичные и внешние ключи, обеспечена ссылочная целостность, наложены ограничения уникальности (госномер автомобиля, артикул запчасти) и проверки доменных значений (положительные цены, годы выпуска, стаж мастеров). Структура базы данных полностью соответствует предметной области автосервиса и гарантирует непротиворечивость и полноту представления информации.

Сформированы DDL-скрипты создания всех таблиц, а также подготовлены DML-скрипты наполнения тестовыми данными. Наполнение выполнено так, чтобы **все контрольные запросы (задания 28–52) возвращали непустые и содержательные результаты**. Для этого включены как типовые записи (базовые марки, модели, запчасти, мастера), так и специально подобранные данные, демонстрирующие крайние и пограничные случаи:

- владельцы с тремя автомобилями одной модели,
- автомобили без единого ремонта,
- заказы с максимальной стоимостью и длительностью,
- поставщики, не покрывающие отдельные категории запчастей,
- мастера с разной активностью в текущем месяце,
- пары автомобилей с одинаковой моделью и годом выпуска.

Для инкапсуляции логики анализа и удобства проверки разработан пакет PL/SQL `autocomplect_queries`, включающий процедуры `query_28`–`query_52` и агрегирующую процедуру запуска. Пакет выводит результаты через `DBMS_OUTPUT`, что позволяет оперативно выполнять регрессионное тестирование и прикладывать машинно-воспроизводимый протокол испытаний.

Проведённые испытания подтвердили корректность заданных ограничений целостности:

- отсутствуют «висячие» ссылки между автомобилями и владельцами;

- все заказы привязаны к существующим автомобилям и мастерам;

- запчасти корректно связаны с категориями и поставками;

- даты выполнения заказов логически согласованы с датами поступления;

- числовые атрибуты (цена, количество, ставка) находятся в допустимых диапазонах.

Запросы с группировками, агрегацией, подзапросами и оконными функциями (статистика по запчастям, анализ поставщиков, отчёт по самым старым автомобилям, выявление владельцев без однофамильцев и т.п.) отрабатывают в полном соответствии с постановкой заданий.

Таким образом, цель проекта достигнута: создана функциональная учебно-практическая база данных автосервиса с чёткой структурой, воспроизводимыми скриптами и полным набором контрольных отчётов. В текущем виде база данных полностью удовлетворяет требованиям задания и демонстрирует корректную работу ключевых сценариев предметной области.