Научная статья
УДК 656.13:004.65
https://doi.org/10.35266/1999-7604-2026-1-2

Язык структурированных запросов для управления данными о транспортных средствах и дорожно-транспортных происшествиях

Дмитрий Юрьевич Зорькин ✉, ***Наталья Васильевна Асанова***
Волгоградский государственный технический университет, Волгоград, Россия

Аннотация. Предметом исследования является применение языка структурированных запросов для систематизации сведений о транспортных средствах, их владельцах и дорожно-транспортных происшествиях в интегрированной информационной среде. Цель работы заключается в разработке и верификации методики проектирования реляционной базы данных, обеспечивающей гарантированную целостность учета и автоматизированный расчет страховых выплат с финансовой точностью. В статье реализована нормализованная структура таблиц посредством DDL-скриптов, включающих ограничения на соответствие данных ГОСТам (на примере регистрационных знаков). Продемонстрирована реализация ключевой бизнес-логики через SQL-запрос, который использует условную логику и агрегирование для корректного учета лимитов ответственности по полису ОСАГО. Проведено сравнение реляционного подхода с нереляционными (NoSQL) и масштабируемыми реляционными (NewSQL) решениями. Сделан вывод о том, что классические SQL-системы являются оптимальными для задач учета, требующих строгой ACID-согласованности и прозрачности вычислений.

Ключевые слова: SQL, реляционная база данных, ДТП, страховые выплаты, целостность данных, DDL, NoSQL

Для цитирования: Зорькин Д. Ю., Асанова Н. В. Язык структурированных запросов для управления данными о транспортных средствах и дорожно-транспортных происшествиях // Вестник кибернетики. 2026. Т. 25, № 1. С. 18–26. https://doi.org/10.35266/1999-7604-2026-1-2.

Original article

Structured query language for data management of vehicles and road traffic accidents

Dmitry Yu. Zorkin ✉, ***Nataliya V. Asanova***
Volgograd State Technical University, Volgograd, Russia

Abstract. The article studies implementation of the structured query language (SQL) in systematizing data on vehicles, their owners, and road traffic accidents (RTAs) within an integrated information environment. The paper aims to develop and validate a method for designing a relational database that ensures accounting integrity and financially precise automated calculation of insurance payments. The work structures table with vehicle registration plates data using DDL scripts, which include constraints for data compliance with state standards. The authors perform the key business logic with an SQL query that applies conditional logic and aggregation for the accurate accounting of liability limits under the compulsory motor third party liability insurance. The study contains a comparison of the relational approach with non-relational (NoSQL) and scalable relational (NewSQL) methods. It is concluded that standard SQL systems are most efficient for accounting tasks requiring strong consistency with ACID properties and calculation transparency.

Keywords: SQL, relational database, RTA, insurance payments, data integrity, DDL, NoSQL

For citation: Zorkin D. Yu., Asanova N. V. Structured query language for data management of vehicles and road traffic accidents. *Proceedings in Cybernetics*. 2026;25(1):18–26. https://doi.org/10.35266/1999-7604-2026-1-2.

ВВЕДЕНИЕ

Автоматизация процессов в транспортной сфере требует надежных инструментов для хранения структурированной информации. Несмотря на развитие NoSQL-решений, реляционные базы данных (RDBMS) остаются стандартом для учетных систем, требующих строгой согласованности данных (ACID), особенно в вопросах финансовых расчетов, таких как страховые выплаты [1, 2]. Цель работы – создание и верификация методики проектирования базы данных для учета (ДТП) и расчета ущерба, исключающей ошибки дублирования и потери данных. Подготовленный выбор данных может быть представлен в приложениях (через формы/отчеты) для обработки [3].

МАТЕРИАЛЫ И МЕТОДЫ

В качестве инструмента реализации выбран язык SQL, обеспечивающий стандартизацию операций выборки и модификации данных. Проектирование выполнялось в три этапа.

1. Концептуальное моделирование путем выделения сущностей «Автомобиль», «Владелец», «Полис ОСАГО», «ДТП».

2. Приведение отношений к третьей нормальной форме (3НФ) для устранения избыточности. Например, данные о владельце вынесены в отдельный справочник, связанный с автомобилем через внешний ключ.

3. Написание DDL-скриптов.

Обоснование использования языка структурированных запросов (SQL)

Язык SQL является общепризнанным стандартом для управления реляционными базами данных (РБД) и обеспечивает строгое разделение функций на Язык определения данных (DDL) и Язык манипулирования данными (DML) [5]. Применение SQL в проектах, связанных с учетом и финансовыми расчетами, обеспечивает ряд ключевых преимуществ, подтвержденных мировым опытом [6, 7].

Стандартизация и переносимость: SQL является индустриальным стандартом, что гарантирует переносимость разработанного решения между различными промышленными СУБД (например, при масштабировании с Access на SQL Server или PostgreSQL) [7].

Гарантия целостности данных: использование DDL позволяет реализовать строгие ограничения целостности (PRIMARY KEY, FOREIGN KEY, CHECK) на уровне сервера, обеспечивая, что в базу данных будут внесены только корректные записи (например, номера автомашин, соответствующие ГОСТу).

Точность вычислений и аналитические возможности: DML позволяет выполнять сложные выборки и агрегирование данных на стороне СУБД. Это минимизирует нагрузку на клиентское приложение и, что критически важно для расчетов ущерба, обеспечивает точность финансовых вычислений за счет использования соответствующих типов данных (DECIMAL) и централизации бизнес-логики.

Для повышения производительности информационной системы при работе с массивами данных применяется индексация ключевых полей (на CarID, RegNumber, AccidentID), что существенно ускоряет процессы фильтрации.

Аналитическая составляющая работы основана на агрегатных функциях (SUM, COUNT), которые используются для вычисления статистических показателей и реализации алгоритма расчета страховой выплаты. В отличие от простого суммирования, расчет выплаты требует учета лимита ответственности по полису ОСАГО, что является примером встраивания сложной бизнес-логики в SQL-запрос.

Для вычисления итоговой суммы, подлежащей выплате с учетом лимита страхования, используется следующий запрос, который представлен на рис. 1.

Этот подход обеспечивает централизацию расчетов и является основой для масштабируемых аналитических систем [8].

РЕЗУЛЬТАТЫ И ИХ ОБСУЖДЕНИЕ

В ходе исследования была разработана и верифицирована методика проектирования реляционной базы данных, ориентированная на строгий учет транспортных средств, владельцев и событий дорожно-транспортных происшествий (ДТП), а также на автоматизированный расчет страховых выплат.

```
SELECT
    c.RegNumber,
    SUM(CASE
        WHEN a.DamageAmount > p.LimitAmount THEN p.LimitAmount
        ELSE a.DamageAmount
    END) AS FinalPayout
FROM Cars c
JOIN Policies p ON c.CarID = p.CarID
JOIN Accidents a ON c.CarID = a.CarID
GROUP BY c.RegNumber;
```

Рис. 1. DDL-скрипт создания структуры всех ключевых таблиц базы данных
Примечание: составлено авторами.

Для описания структуры данных использовался язык определения данных (DDL). Предложенная структура соответствует требованиям третьей нормальной формы (3НФ), что устраняет избыточность и обеспечивает целостность данных, включая те сущности, которые были неполно или некорректно представлены в исходном материале (например, данные о владельцах).

Особое внимание уделено ограничениям целостности (Constraints).

1. Введены отдельные сущности для Владельцев (Owners), Страховых полисов (Policies) и ДТП (Accidents), которые корректно связаны между собой.

2. Применены ограничения CHECK для соответствия полей нормативно-правовым требованиям (например, для формата регистрационного знака по ГОСТ Р 50577–2018).

3. Для финансовых полей (LimitAmount, DamageAmount) установлен тип данных DECIMAL (15, 2), обеспечивающий строгую точность до копеек.

Полная, исправленная структура всех ключевых таблиц, заменяющая некорректные графические иллюстрации, представлена на рис. 2.

После успешного определения структуры данных выполняется наполнение таблиц тестовыми записями с помощью оператора INSERT – основного элемента Языка манипулирования данными (DML). Приведенный на рис. 2 фрагмент демонстрирует, как данные о всех сущностях (автомобилях, владельцах, полисах и ДТП) вносятся в базу, проходя автоматическую валидацию через ограничения, определенные на рис. 3.

Для извлечения, анализа и обработки данных применяется команда SELECT. Разработанная модель требует обязательного объединения сущностей (JOIN) для получения полной картины инцидента, связывая информацию о ДТП с конкретным автомобилем и его актуальным страховым полисом.

Ключевым результатом работы является применение SELECT для реализации финансовой бизнес-логики. С его помощью, а также с использованием агрегатных функций и условной логики (CASE WHEN), реализуется расчет страховой выплаты с учетом лимита ответственности по полису ОСАГО.

В качестве примера реализации сложной бизнес-логики рассмотрим автоматизированный расчет суммы, подлежащей выплате в результате ДТП. Расчет страховой выплаты – является нетривиальной задачей, требующей сравнения заявленного ущерба с лимитом ответственности, установленным полисом ОСАГО.

```
-- Таблица 1: Cars (Автомобили)
CREATE TABLE Cars (
    CarID INT PRIMARY KEY IDENTITY(1,1),
    RegNumber NVARCHAR(9) NOT NULL
        CHECK (RegNumber LIKE '[А-Я] [0-9][0-9][0-9] [А-Я][А-Я]
[0-9][0-9]'),
    Model NVARCHAR(100) NOT NULL,
    VIN NVARCHAR(17) UNIQUE
);

-- Таблица 2: Owners (Владельцы) -- Заменяет Рис. 3, 4, 5, 6
CREATE TABLE Owners (
    OwnerID INT PRIMARY KEY IDENTITY(1,1),
    FullName NVARCHAR(200) NOT NULL,
    PassportData NVARCHAR(100) UNIQUE
);

-- Таблица 3: Policies (Полисы ОСАГО)
CREATE TABLE Policies (
    PolicyID INT PRIMARY KEY IDENTITY(1,1),
    CarID INT FOREIGN KEY REFERENCES Cars(CarID),
    OwnerID INT FOREIGN KEY REFERENCES Owners(OwnerID),
    PolicyNumber NVARCHAR(20) UNIQUE NOT NULL,
    LimitAmount DECIMAL(15, 2) NOT NULL
);

-- Таблица 4: Accidents (ДТП)
CREATE TABLE Accidents (
    AccidentID INT PRIMARY KEY IDENTITY(1,1),
    PolicyID INT FOREIGN KEY REFERENCES Policies(PolicyID),
    AccidentDate DATE NOT NULL,
    DamageAmount DECIMAL(15, 2) NOT NULL
);
```

Рис. 2. DDL-скрипт создания структуры всех ключевых таблиц базы данных
Примечание: составлено авторами

```
-- DML: Наполнение таблицы Cars (Автомобили)
INSERT INTO Cars (RegNumber, Model, VIN) VALUES
('A 123 AA 34', 'Lada Vesta SW', 'X7L00000000000001');

-- DML: Наполнение таблицы Owners (Владельцы)
INSERT INTO Owners (FullName, PassportData) VALUES
('Иванов Иван Иванович', '1234 567890');

-- DML: Наполнение таблицы Policies (Полисы)
INSERT INTO Policies (CarID, OwnerID, PolicyNumber, LimitAmount) VALUES
(1, 1, '0123456789', 400000.00);

-- DML: Наполнение таблицы Accidents (ДТП)
INSERT INTO Accidents (PolicyID, AccidentDate, DamageAmount) VALUES
(1, '2025-10-20', 150500.50);
```

Рис. 3. Фрагмент DML-скрипта для наполнения всех ключевых таблиц
Примечание: составлено авторами.

Для решения этой задачи используется агрегатный запрос, объединяющий данные из таблиц Accidents, Policies, Cars и Owners. Ключевой элемент запроса – условный оператор CASE WHEN, который позволяет реализовать логику ограничения выплаты страховым лимитом. Применение агрегатной функции SUM обеспечивает корректное суммирование выплат по каждому автомобилю (при необходимости).

Финальный расчетный запрос, демонстрирующий применение оператора JOIN для связывания сущностей и условной логики для вычислений, представлен на рис. 4. Полученный результат (FinalPayout) является непосредственным итогом работы информационной системы и может быть использован в отчете «Страховые выплаты».

Приведенный на рис. 4 запрос подтверждает, что разработанная структура позволяет не только хранить, но и эффективно анализировать данные. Реализация сложной логики на уровне сервера БД обеспечивает централизацию бизнес-правил и гарантию точности финансовых расчетов, что является ключевым преимуществом перед ненормализованными решениями.

Сравнение с альтернативными подходами к управлению данными

Помимо реляционных СУБД, в последние годы получили развитие альтернативные модели хранения данных. Сравним реляционный подход (SQL), Нереляционные Базы Данных (NoSQL) и Новый Структурированный Язык Запросов (NewSQL) подходы в контексте транспортной предметной области.

Реляционные СУБД (SQL). Классические SQL-решения обеспечивают формальную согласованность данных: Атомарность, Согласованность, Изолированность, Долговечность (АСИД-транзакции) и позволяют выразительно описывать сложные связи через JOIN [9]. Как отмечено в обзорах по транспортным данным, реляционные СУБД «широко

```
SELECT
    c.RegNumber,
    o.FullName AS OwnerName,
    p.LimitAmount AS PolicyLimit,
    a.DamageAmount AS DeclaredDamage,
    -- Применение условной логики для расчета фактической выплаты
    SUM(CASE
        -- Если заявленный ущерб больше лимита, выплачивается лимит
        WHEN a.DamageAmount > p.LimitAmount THEN p.LimitAmount
        -- Иначе, выплачивается заявленный ущерб
        ELSE a.DamageAmount
    END) AS FinalPayout
FROM Accidents a
JOIN Policies p ON a.PolicyID = p.PolicyID
JOIN Cars c ON p.CarID = c.CarID
JOIN Owners o ON p.OwnerID = o.OwnerID
GROUP BY
    c.RegNumber, o.FullName, p.LimitAmount, a.DamageAmount
ORDER BY c.RegNumber;
```

Рис. 4. SQL-запрос для расчета страховых выплат с учетом лимита ОСАГО
Примечание: составлено авторами.

используются для управления различными формами транспортных данных» [9]. Они позволяют легко оперировать пространственными и временными атрибутами (например, через расширения PostGIS) [9]. К достоинствам SQL-модели относится строгость и надежность схемы [9].

Однако у традиционных реляционных моделей есть ограничения: при очень больших потоках данных (Big Data) и быстро меняющихся структурах (например, динамических топологиях дорожных сетей) реляционные таблицы могут требовать сложной перестройки схемы. Указано, что «реляционное моделирование недостаточно гибко при сложных и эволюционирующих структурах данных», что особенно важно для комплексных транспортных сетей [9]. Дополнительное хранилище (NoSQL, хранилища Data Lake) внедряется для оптимизации таких случаев, но при этом возрастает сложность интеграции данных в реальном времени. Таким образом, традиционные SQL-базы хорошо подходят для статичных структурированных данных (регистрационные учеты, финансовые отчеты и т. п.), но могут испытывать трудности при обработке в режиме «реального времени» и неструктурированной информации без адаптации инфраструктуры [9].

Нереляционные СУБД (NoSQL). NoSQL-решения включают различные типы (документные, графовые, ключ-значение,

колоночные) и ориентированы на гибкость схемы и масштабируемость [9]. Они часто соответствуют принципам Доступный, Мягкое состояние, Постепенно согласованный (BASE) [9] и хорошо подходят для неструктурированных и полуструктурированных данных – например, больших потоков телеметрии, логов, JSON-документов маршрутов и др. В транспортной сфере NoSQL может применяться для хранения сетевых графов дорог в графовых Баз Данных (БД), потоковых данных от датчиков (ключ-значение), данных справочников маршрутов (JSON в документных баз данных) [10]. Как отмечают исследователи, миграция смарт-городских решений на NoSQL оправдана при «больших объемах и Big Data» [10]. К примеру, в проектах умного города Документ-Ориентированная Система Управления Базами Данных (MongoDB) и Удаленный Сервер Словарей (Redis) показали высокую производительность и масштабируемость на гибких данных [10].

Но есть и минусы: отказ от строгой ACID-согласованности может быть критичным для финансовых расчетов или юридически значимых операций (в страховании ДТП, регистрациях). Кроме того, в NoSQL сложнее выполнять сложные JOIN-операции по нескольким таблицам, а поддержание целостности данных обычно ложится на приложение. Таким образом, NoSQL-базы выгодны при необходимости обработки разноформатных больших объемов данных (обходя необходимость жестких схем), но могут уступать в связности и согласованности для традиционных учетных задач.

NewSQL-системы. NewSQL – сравнительно новый класс СУБД, который сочетает привычный SQL-интерфейс с современными технологиями масштабирования [11]. Такие системы сохраняют полную совместимость с SQL (нет необходимости учить новый язык), но используют распределенные и in-memory архитектуры для достижения высокой производительности [11]. Примеры NewSQL: VoltDB, NuoDB, CockroachDB. Эти системы обещают горизонтальную масштабируемость без потери ACID [11]. В контексте транспортных приложений NewSQL может быть актуален для распределенных облачных систем с экстремальными требованиями к пропускной способности (например, обработка миллиарда записей телеметрии в реальном времени). Однако NewSQL – более сложная в развтртывании технология, и ее преимущества проявляются в тех случаях, когда традиционные СУБД уже не справляются с нагрузкой.

В целом выбор подхода зависит от задач: для большинства классических систем учета ДТП реляционные СУБД с SQL остаются оптимальным вариантом благодаря гибкости запросов и надежности схемы. NoSQL может дополнять решения при работе с потоковыми или графовыми данными, а NewSQL обеспечивает перспективу масштабирования при росте объемов. Мы видим, что нет единого решения на все случаи: комбинация реляционной модели (для основного учета), и при необходимости, специальных хранилищ (NoSQL) является разумной стратегией.

ЗАКЛЮЧЕНИЕ

В результате проведенного исследования была разработана и верифицирована методика проектирования реляционной базы данных для интегрированного учета транспортных средств, владельцев и дорожно-транспортных происшествий.

Показано, что применение Языка определения данных (DDL), включая ограничения CHECK на уровне сервера, обеспечивает строгое соблюдение нормативных требований к вводимым данным (например, формату регистрационного номера по ГОСТу), достичь которого невозможно в нереляционных или файловых системах.

На основе языка манипулирования данными (DML) и агрегатных функций был реализован механизм автоматизированного расчета страховых выплат. Тип данных DECIMAL и условная логика CASE WHEN обеспечили финансовую точность расчетов. Это гарантировало корректный учет лимитов ответственности по полису ОСАГО.

Проведенное критическое сравнение подходов SQL, NoSQL и NewSQL показало, что реляционная модель – наиболее надежное и оптимальное решение. Именно реляционная модель гарантирует транзакционную целостность (ACID).

Список источников

1. Silberschatz A., Korth H. F., Sudarshan S. Database system concepts. 7th ed. New York : McGraw-Hill, 2019. 1376 p.
2. Coronel C., Morris S. Database systems: Design, implementation, & management. 13th ed. Boston : Cengage Learning, 2018. 816 p.
3. Таланов В. М., Рунков М. В., Ерофеев Г. В. Пример создания базы данных в системе управления базами данных MySQL // XLVI Огарёвские чтения : материалы научной конференции, 6–13 декабря 2017 г., г. Саранск. В 3-х частях. Саранск : Национальный исследовательский Мордовский государственный университет им. Н. П. Огарёва, 2018. Часть 1. С. 219–224.
4. Коннолли Т., Бегг К. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / пер. с англ. М. : Вильямс, 2017. 1440 с.
5. Грофф Дж. Р., Вайнберг П. Н., Оппель А. Дж. SQL: Полное руководство / пер. с англ. 3-е изд. СПб. : Диалектика, 2019. 960 с.
6. Соколин Д. Д., Тимохович А. С. Методы комплексного обеспечения безопасности SQL-сервера от атак типа SQL-инъекции // Academy. 2017. № 3 (18). С. 7–9.
7. Gondal S., Davkhure V., Shivalkar A. et al. No SQL – Not obligatory SQL (Natural language to SQL conversion) // International Research Journal of Modernization in Engineering Technology and Science. 2023. Vol. 5, no. 4. P. 1304–1308. https://www.doi.org/10.56726/IRJMETS35725.
8. Буняева Е. В., Романова А. Г. Различия диалектов языка SQL в СУБД Microsoft SQL Server и Postgree SQL // Развитие науки и практики в глобально меняющемся мире в условиях рисков (шифр – МКРНП) : сборник материалов XXIX Международной научно-практической конференции, 28 июня 2024 г., г. Москва, Москва : Издательство «Экономическое образование», 2024. С. 232–241.
9. Bellini P., Bilotta S., Collini E. et al. Data sources and models for integrated mobility and transport solutions // Sensors. 2024. Vol. 24, no. 2. P. 441. https://doi.org/10.3390/s24020441.
10. Gusti Ayu Mas Ekayanti, Dewa Ayu Deby Cintiya, Putu Yoga Suartana et al. Comparison of SQL sus, SQL ninja, and the mole tools in implementing SQL injection // Jurnal Informatika Teknologi dan Sains. 2022. Vol. 4, no. 4. P. 478–482. https://doi.org/10.51401/jinteks.v4i4.2201.
11. Pina E., Sá F., Bernardino J. NewSQL databases assessment: CockroachDB, MariaDB Xpand, and VoltDB // Future Internet. 2023. Vol. 15, no. 1. https://doi.org/10.3390/fi15010010.

References

1. Silberschatz A., Korth H. F., Sudarshan S. Database system concepts. 7th ed. New York: McGraw-Hill; 2019. 1376 p.
2. Coronel C., Morris S. Database systems: Design, implementation, & management. 13th ed. Boston: Cengage Learning; 2018. 816 p.
3. Talanov V. M., Runkov M. V., Erofeev G. V. Primer sozdaniya bazy dannykh v sisteme upravleniya bazami dannykh MySQL. In: *Proceedings of the Scientific Conference "XLVI Ogarevskie chteniya"*, December 6–13, 2017, Saransk. In 3 pts. Saransk: National Research Ogarev Mordovia State University; 2018. Pt. 1. p. 219–224. (In Russ.).
4. Connolly T., Begg C. Databases: Design, implementation, and maintenance. Theory and practice. Trans. Moscow: Williams; 2017. 1440 p. (In Russ.).
5. Groff J. R., Weinberg P. N., Oppel A. J. SQL: The complete reference. Trans. 3rd ed. St. Petersburg: Dialektika; 2019. 960 p. (In Russ.).
6. Sokolin D. D., Timokhovich A. S. Metody kompleksnogo obespecheniya bezopasnosti SQL-servera ot atak tipa SQL-inektsii. *Academy*. 2017;(3):7–9. (In Russ.).
7. Gondal S., Davkhure V., Shivalkar A. et al. No SQL – Not obligatory SQL (Natural language to SQL conversion). *International Research Journal of Modernization in Engineering Technology and Science*. 2023;5(4):1304–1308. https://www.doi.org/10.56726/IRJMETS35725.
8. Bunyaeva E. V., Romanova A. G. Differences in SQL language dialects in Microsoft SQL Server and Postgre SQL DBMS. In: *Proceedings of the 29th International Research-to-Practice Conference "Razvitie nauki i praktiki v globalno menyayushchemsya mire v usloviyakh riskov (shifr – MKRNP)"*, June 28, 2024, Moscow. Moscow: "Economic Education" Publishing House; 2024. p. 232–241. (In Russ.).
9. Bellini P., Bilotta S., Collini E. et al. Data sources and models for integrated mobility and transport solutions. *Sensors*. 2024;24(2):441. https://doi.org/10.3390/s24020441.
10. Gusti Ayu Mas Ekayanti, Dewa Ayu Deby Cintiya, Putu Yoga Suartana et al. Comparison of SQL sus, SQL ninja, and the mole tools in implementing SQL injection. *Jurnal Informatika Teknologi dan Sains*. 2022;4(4):478–482. https://doi.org/10.51401/jinteks.v4i4.2201. (In Indonesian).
11. Pina E., Sá F., Bernardino J. NewSQL databases assessment: CockroachDB, MariaDB Xpand, and VoltDB. *Future Internet*. 2023;15(1). https://doi.org/10.3390/fi15010010.

Информация об авторах

Д. Ю. Зорькин – преподаватель;
https://orcid.org/0009-0002-3875-9285,
mosh285@gmail.com✉
Н. В. Асанова – кандидат технических наук, доцент;
https://orcid.org/0000-0003-3781-7017,
natali_as@mail.ru

About the authors

D. Yu. Zorkin – Lecturer;
https://orcid.org/0009-0002-3875-9285,
mosh285@gmail.com✉
N. V. Asanova – Candidate of Sciences (Engineering), Docent;
https://orcid.org/0000-0003-3781-7017,
natali_as@mail.ru