

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГБОУ ВО МГТУ «СТАНКИН»)**

|  |  |
| --- | --- |
| **Институт** информационных технологий | **Кафедра** информационных систем |

**Отчет по самостоятельной работе**

по дисциплине «**Управление данными**»

на тему: Проектирование БД гостиницы

**Студент**  **Грошев В.П.** группа ИДБ–20–05

подпись

**Руководитель Быстрикова В. А.** старший преподаватель

подпись

Москва 2022 г.

**Глава 1. Анализ предметной области**

Предметную область можно определить как сферу человеческой деятельности, выделенную и описанную согласно установленным критериям. В описываемое понятие должны входить сведения об ее элементах, явлениях, отношениях и процессах, отражающих различные аспекты этой деятельности.

Одна из первых задач, с решением которых сталкивается разработчик программной системы – это изучение, осмысление и анализ предметной области. Дело в том, что предметная область сильно влияет на все аспекты проекта: требования к системе, взаимодействие с пользователем, модель хранения данных, реализацию и т.д. Анализ предметной области позволяет выделить ее сущности, определить первоначальные требования к функциональности и определить границы проекта.

Предметной областью данной самостоятельной работы является деятельность гостиницы, предоставляющей свои услуги посетителям.

Сущность предоставления услуги размещения состоит в том, что, с одной стороны, в пользование предоставляются специальные помещения (гостиничные номера), с другой стороны, предоставляются услуги, выполняемые непосредственно персоналом гостиницы: услуги портье по приему и оформлению гостей, услуги горничных по уборке гостиничных номеров и т.д.

Гостиничные номера являются основным элементом услуги размещения. Это многофункциональные помещения, предназначенные для отдыха, сна, работы проживающих гостей. Важнейшей их функцией является обеспечение возможности сна. Значимость других функций гостиничных номеров, прежде всего, зависит от назначения гостиницы и потребностей гостей. Например, в гостиницах делового назначения. Именно поэтому задачей данной работы является автоматизация и упрощение работы сотрудников гостиницы. Для этого целесообразно использовать базы данных.

База данных – совокупность специальным образом организованных данных, хранимых в памяти вычислительной системы, отображающих состояние объектов и их взаимосвязей в предметной области.

Существует множество готовых программных продуктов для выполнения задач в этой области. В качестве первого примера рассмотрим программный продукт «1С: Предприятие 8. Отель» (https://solutions.1c.ru/catalog/hotel/features). Программный продукт предназначен для цифровизации работы сотрудников гостиницы от управления номерами до регистрации посетителей. Гостиничная программа представляет собой комплексное программное решение для автоматизации работы гостиницы. Основными решаемыми задачами являются учет клиентов, учет проживания, отчет о стоимости номеров.

К основным функциями программы относятся:

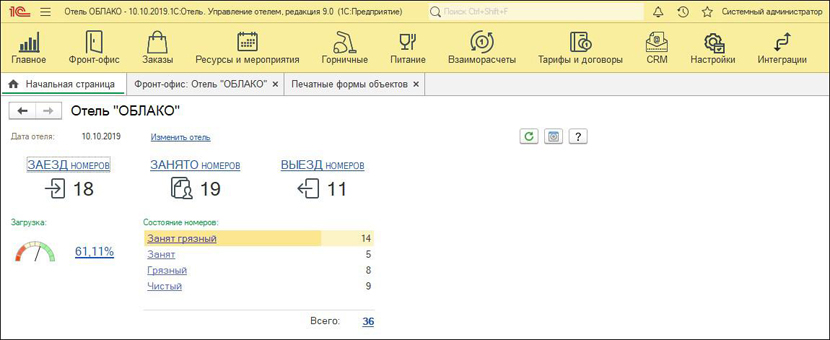
* Отображение ключевых показателей на главном экране (рис. 1.1).

Рис. 1.1 Главный экран, отображающий ключевые показатели

* Визуализация интерактивной графической карты (рис. 1.2).

Интерактивная графическая карта номерного фонда подходит для визуальной оценки загрузки номеров и позволяет в режиме drag & drop переносить брони и оптимизировать загрузку. Масштаб карты подстраивается по разрешение экрана и среднюю продолжительность проживания – от часа до месяца. Все операции фронт-офиса доступны из одного окна – поиск и поселение брони, бронирование, смена статусов номеров, продление и переселение гостей.

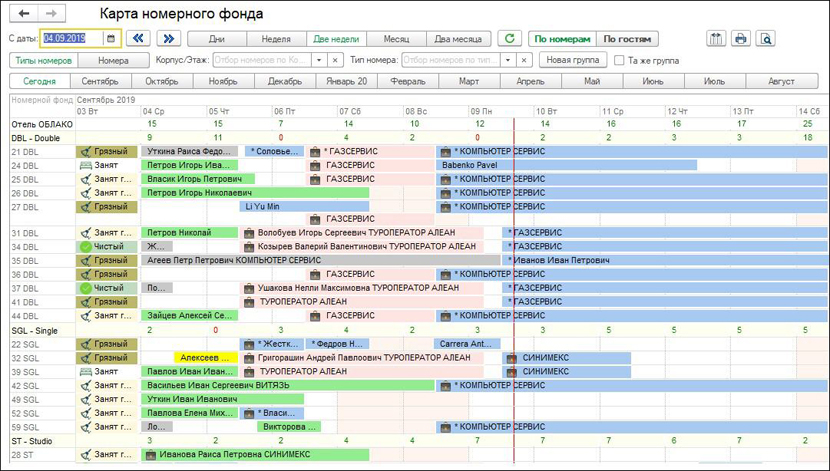


Рис. 1.2 Интерактивная графическая карта

* Сканер паспортов

Сканер паспортов. Подключение сканера паспортов к системе ускоряет процесс оформления гостей, облегчает работу ресепшн, экономит бумагу и картриджи при копировании паспортов. 1С: Отель сохранит копию паспорта, чтобы администраторы могли позже, в свободное время, обработать их в едином журнале. Обработанные документы автоматически заполняют и обновляют профиль гостя, а также записи в журнале регистрации иностранных граждан. При подключении систем распознавания таких как ABBYY Passport Reader SDk или PassportBox (приобретаются отдельно) заполнение всех данных клиента происходит автоматически.

* Экспресс заселение. Для ускорения оформления гостя при заезде используется механизм экспресс-заселения по QR Коду. При считывании QR-кода из подтверждения бронировании происходит автоматический поиск брони, заселение и выдача ключа. Таким образом время оформления гостя сокращается до 5 секунд. Такой код в подтверждении получают гости, у которых бронь заранее оплачена и внесены паспортные данные гостя.
* Контроль взаиморасчетов (рис. 1.3).

Контроль взаиморасчетов. Система позволяет быстро и удобно вести взаиморасчеты с гостями и контрагентами. С помощью гибких настроек правил начисления счета разделяются по видам услуг, срокам, суммам и т.д. Для контроля задолженности программа учитывает глубину кредита, и сумму нефискальных предавторизаций по карте гостя. Простой механизм оформления авансов и проведения операций через онлайн ККМ (54-ФЗ) сокращает число ошибок и время оформления операций с гостем.

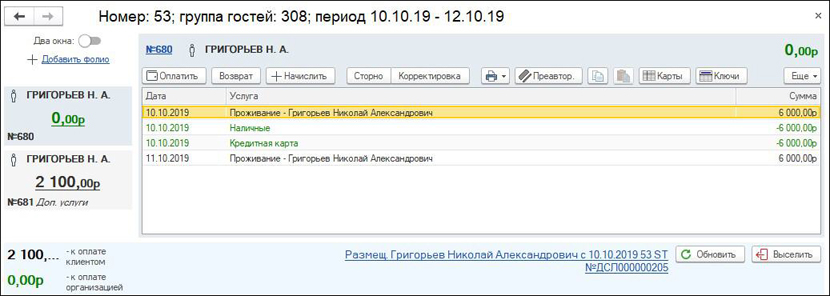


Рис. 1.3 Контроль взаиморасчетов

* Индивидуальное и групповое бронирование (рис. 1.4). Новая форма наличия номеров позволяет быстро оценить загрузку, проверить цены, рассчитать и забронировать в 2 клика как индивидуальную бронь, так и групповую. В результате клиент может получить счет с подписью и печатью в течении минуты.

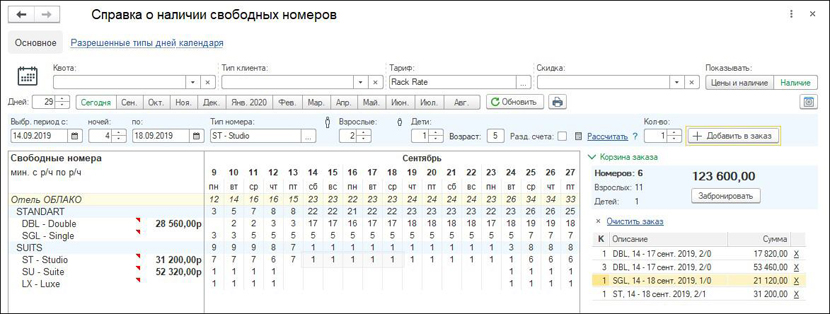


Рис. 1.4 Индивидуальное и групповое бронирование

* Управление загрузкой и квотами номеров (рис.1.5). Использование механизма квот позволяет выделить часть номерного фонда гостиницы в распоряжение агента (туроператора). При этом номера, выделенные в квоту, можно снять с продажи и заселять в эти номера гостей только по заявкам агента. Система квот позволяет создавать "мягкие" и "жесткие" квоты. Дата освобождения квоты контролируется автоматически, что повышает возможности продать номер напрямую.

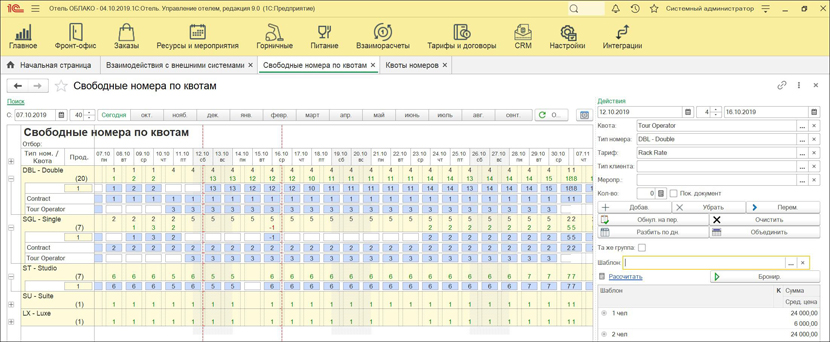


Рис. 1.5 Управление загрузкой и квотами номеров

* Динамические тарифы (рис. 1.6). Система управления тарифами позволит реализовать гибкие схемы по продаже. Удобный новый календарь управления уровнем цены и ограничениями по тарифу позволит вести более гибкую ценовую политику. Кроме того, система поддерживает автоматическое повышение цены в зависимости от % загрузки отеля или категории.

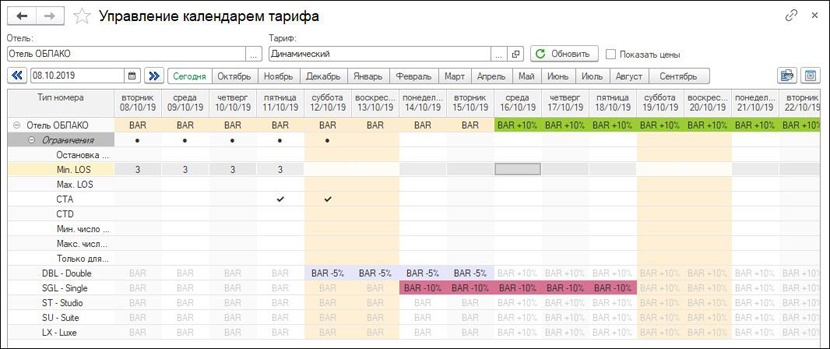


Рис. 1.6 Динамические тарифы

* Управление скидками. Система поддерживает работу с простыми и накопительными скидками. Есть возможность задать различный процент скидки для разных наборов услуг, что позволит отелю построить программу лояльности с привлекательными, но не разорительными условиями.

Накопительные скидки автоматически учитывают сумму продаж, число заездов или число ночей по клиентам и применяют соответствующий уровень.

* Автоназначение скидок. Система проследит за суммой продаж, ночей или заездов по клиенту и автоматически сменит тип скидки у гостя. При этом менеджер получит уведомление со списком гостей и их новыми уровнями в программе лояльности.
* Бонусы. Система накопит бонусы по правилам отеля. Чтобы их потратить гостю нужно вернуться, что повышает возвратность и лояльность гостей. 1С: Отель не только учитывает бонусы, но и предоставляет интерфейс для использования бонусов во внешних системах, что позволит построить полную и удобную для гостя систему.
* Интеграция и единый счет гостя. Обмен данными с [1С:Салон красоты](https://solutions.1c.ru/catalog/beauty-salon/features), [1C:SPA-Салон](https://solutions.1c.ru/catalog/spa-salon/features), [1С:Фитнес клуб](https://solutions.1c.ru/catalog/fitness/features) и [1С:Ресторан](https://solutions.1c.ru/catalog/restaurant/features). В поставку включены веб и http сервисы, которые помогут интегрировать все системы в единое информационное пространство и предоставить гостю возможность оплатить любой свой заказ в одном месте – на ресепшн отеля.

POS-система ресторана или SPA-комплекса получает данные о клиенте, его скидках, лимитах и балансе и учитывает при оформлении заказа. Обратно в 1С: Отель из POS-системы автоматически предается на счет гостя, включая детализацию заказа по позициям. Упрощение расчетов и удобство для гостя повышает средний чек в отеле и, в тоже время, контролирует своевременную оплату всех счетов.

* Заказ трансфера. Новая подсистема "Заказы" позволит не просто заказать услугу трансфера, но и автоматизировать соответствующую службу.

На рисунке 1.7 изображен заказ, у которого отслеживается статус заказа вплоть до исполнения.

Также поддерживаются различные сценарии оплаты – на номер, наличными, водителю. Заказ сохранит всю необходимую информацию, включая место встречи и количество детских кресел.

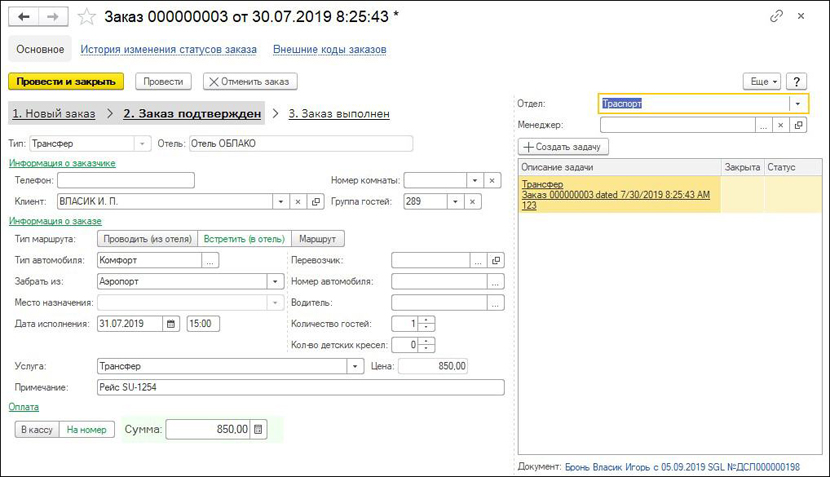


Рис. 1.7 Заказ

* Управление статусами номеров. Для отображения занятости номеров и состояния уборки используются статусы номеров. Статусы также необходимы при планировании и распределении работ по номерам.

Изначально номер "свободный, чистый". В день заезда гостя на несколько дней статус номера меняется на "занятый, чистый". На следующий день номер становится грязным, и он по-прежнему занят гостями – статус "занятый, грязный". Горничная проводит уборку, что возвращает номер в состояние "занятый, чистый". Так продолжается каждый день, пока гости живут в номере. В день выезда гостя статус изменится на "свободный, грязный". Горничная проводит уборку ставит статус "свободный, чистый".

На рисунке 1.8 удобный отчет для горничных, который показывает какие работы в каком номере нужно выполнить. Это поможет даже небольшим отелям, где в штате нет супервайзера для контроля работ. Также система позволяет планировать работу и выдавать задания индивидуально для каждого сотрудника.

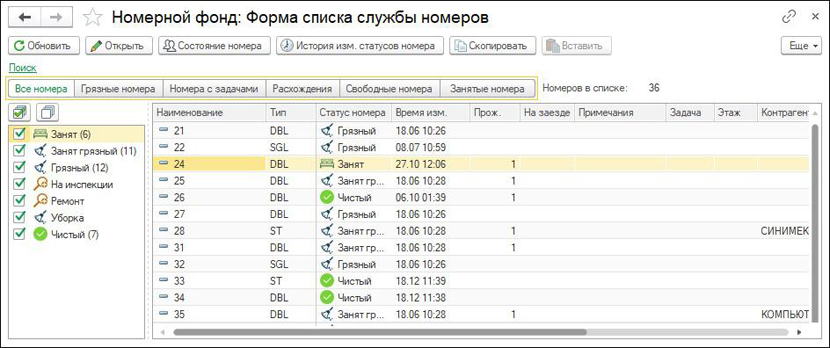


Рис. 1.8 Отчет для горничных

* Распределение работ (рис. 1.9). Специальное рабочее место Супервайзера используется для назначения работы горничным, и печати персонального плана работ на день.

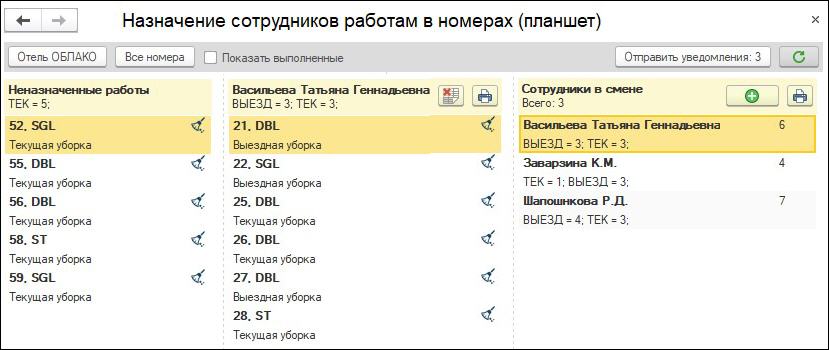


Рис. 1.9 Распределение работ

* Учет материалов. По каждому виду работ система рассчитает плановый и фактический расход материалов и номенклатуры. Нормативы задаются по видам работ на категорию номера или на номер. Специальный отчет рассчитает какой должен быть оборот материалов и принять правильные решения по мотивации горничных.
* Контроль оплаты счетов (рис. 1.10). Счета на оплату выставляются очень быстро и просто как на группу гостей, так и на отдельный номер или мероприятие. По каждому счету система отслеживает статус и срок оплаты.

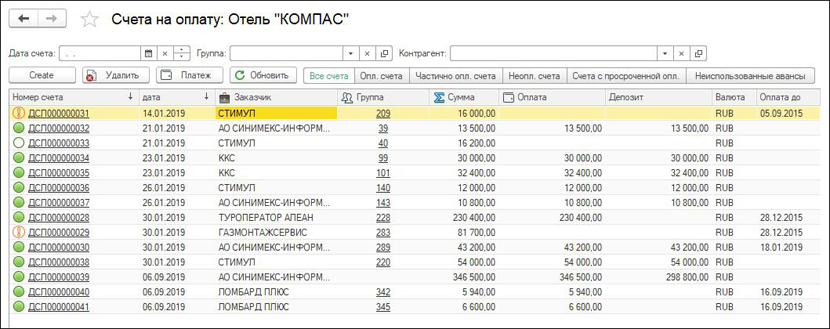


Рис. 1.10 Контроль оплаты счетов

* Интеграция с бухгалтерией. Обмен данными с [1С:ERP Управление предприятием](http://v8.1c.ru/erp/) , [1С:Комплексная автоматизация, редакция 2](http://v8.1c.ru/ka2/) и [1С:Бухгалтерия](https://v8.1c.ru/buhv8/). Все счета выгружаются в бухгалтерию, обратно загружаются платежи. Таким образом менеджер отеля получает своевременную информацию об оплате счета, и процесс контроля не отнимает много времени.

Рассмотрим второй программный продукт к**онфигурация "Простая гостиница" (https://prostoysoft.ru/Hotel.htm)** предназначена для автоматизации гостиниц, отелей, баз отдыха, санаториев и пансионатов. Данная программа значительно улучшит и облегчит работу администратора, благодаря возможности контроля над общей ситуацией в отеле. Конфигурация позволяет объединить все процессы, связанные с бронированием гостей, расчетами с посетителями, туроператорами и агентами, управлением персоналом и многое другое. Программа легко и быстро настраивается под конкретные требования заказчика.

Гибкая структура базы данных дает возможность создавать новые таблицы, отчеты, графики, добавлять поля, задавать списки и многое другое.

К основным функции программы относятся:

* Учет посетителей гостиницы или отеля (рис. 1.13);
* Хранение контактных данных посетителей (рис. 1.13);
* Справочник всех номеров (рис. 1.14);
* Бронирование номеров (рис. 1.12);
* Размещение гостей (рис. 1.11);
* Контроль и планирование выполнения работ в номерах;
* Учет электронных пропусков;
* Управление тарифами по дням недели, по сезонам;
* Учет дополнительных услуг отеля: сейф, телефонные звонки, заказ такси, сауна и др.;
* Планирование мероприятий и банкетов;
* Контроль взаиморасчетов с гостями;
* Хранение информации о сотрудниках, настройка персональных прав доступа к системе для сотрудников (рис. 1.16);
* Возможность формирования ежемесячной заработной платы для сотрудников (рис. 1.15);
* Возможность импорта и экспорта данных;

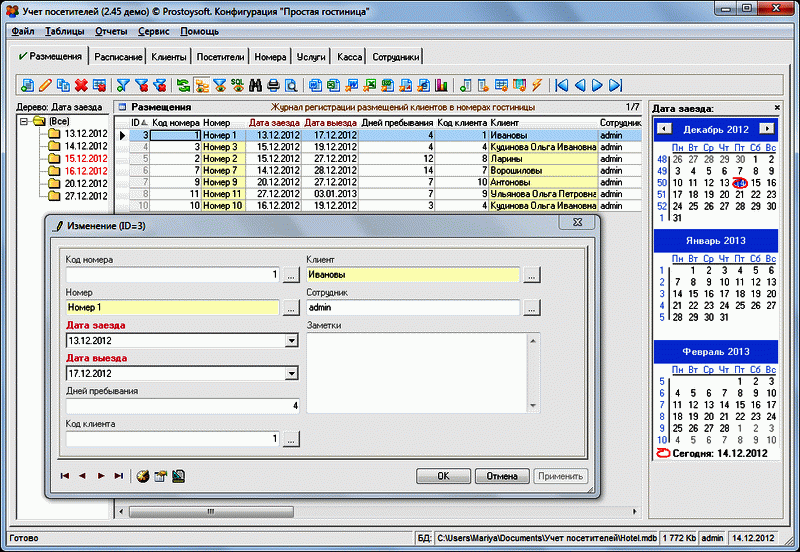


Рис. 1.11 Контроль размещения гостей

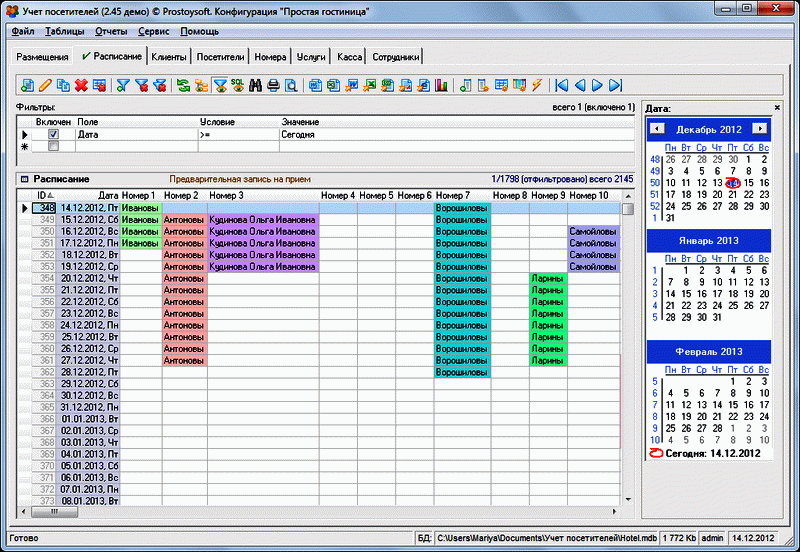


Рис. 1.12 График размещений

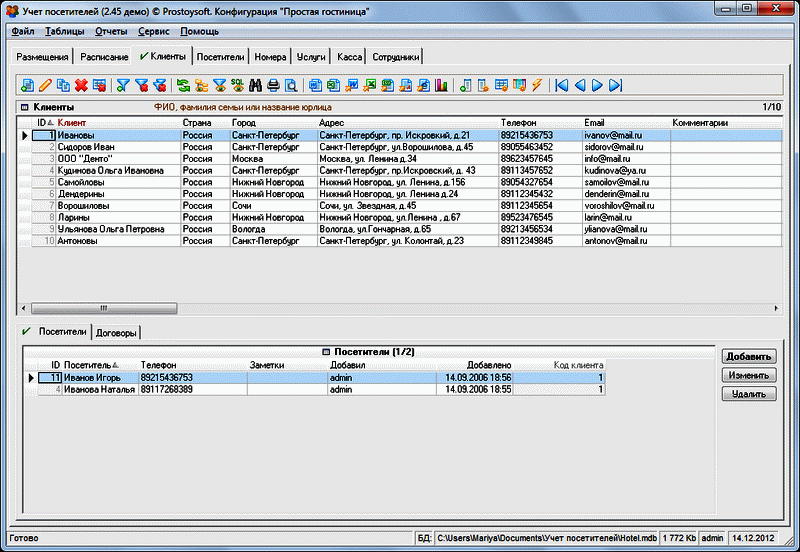


Рис. 1.13 Учет клиентов гостиницы, их контактные данные

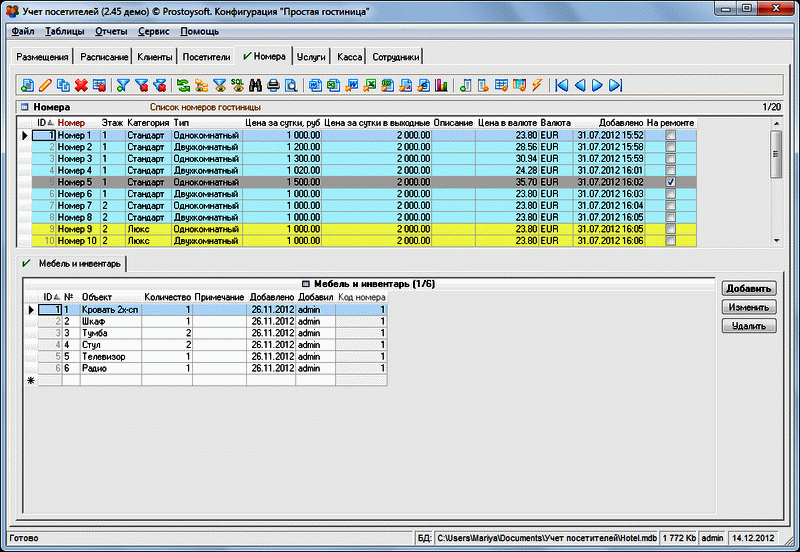


Рис. 1.14 Справочник номеров

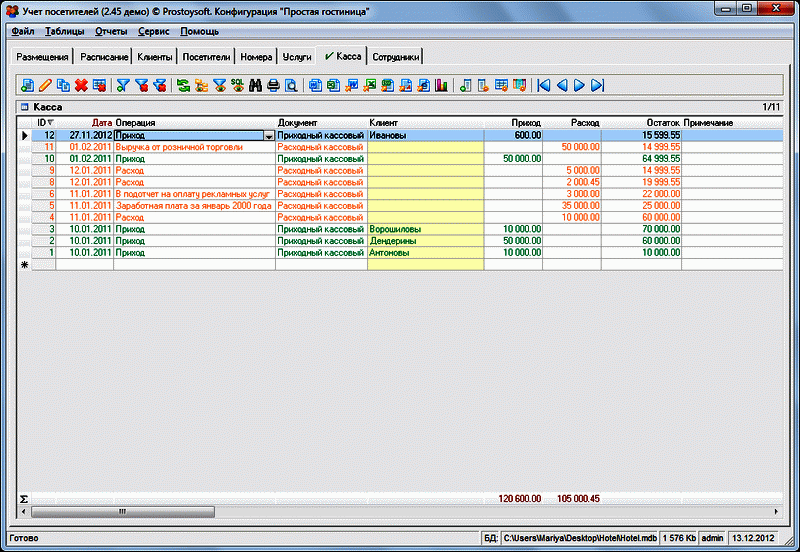


Рис. 1.15 Касса: контроль движения денежных средств

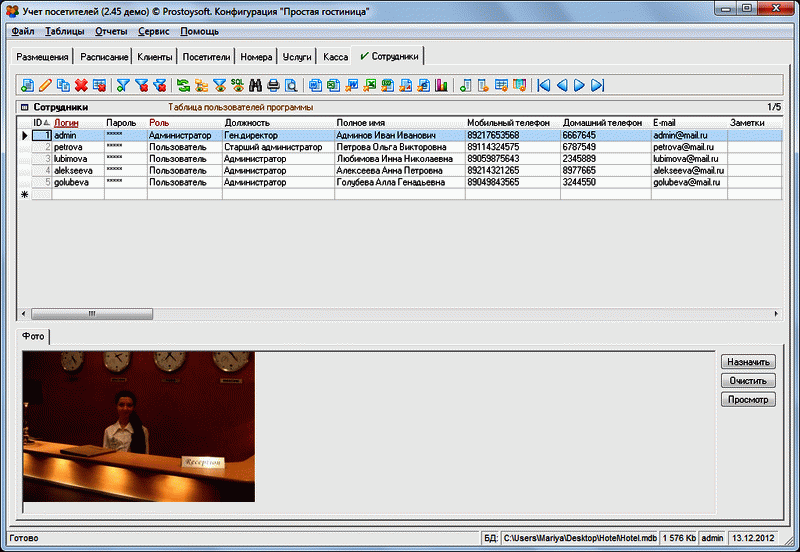


Рис. 1.16 Перечень сотрудников компании и пользователей с указаниями параметров доступа к программе

Для всех рассмотренных программных продуктов можно выделить общие функции:

* Учет посетителей
* Учет номеров
* Управление ценами

Особенностью программы «1С: Предприятие 8. Отель» является наличие большого количества функции, которые могут позволить ускорить процесс ведения базы данных, а также возможность визуализировать данные, что дает простой анализ ситуации, также стоит учесть, что есть интеграции с другими продуктами 1С.

Особенностью Конфигурации "Простая гостиница" — это простота и легкость в освоении в отличии от 1С, который наделен множеством разных функций. Из своих функций имеет возможность хранить информацию о сотрудниках, настройка персональных прав доступа к системе для сотрудников

После проведения анализа предметной области был выделен перечень функций, которые будут реализованы в работе.

* Управление состоянием номеров
* Управление бронированием
* Регистрация посетителей
* Оформление услуг клиента
* Учет постельного белья и полотенец
* Распределение обязанностей
* Просмотр графика сотрудников
* Управление наймов сотрудников
* Контроль оплаты счетов
* Составление отчетов для руководства

**Глава 2. Концептуальное проектирование**

Концептуальное проектирование базы данных — это процесс создания модели используемой на предприятии информации, не зависящей от любых физических аспектов ее представления. Концептуальное проектирование заключается в формализованном описании предметной области, это описание должно быть таким, чтобы, с одной стороны, можно было проанализировать корректность схемы разрабатываемого проекта БД, с другой стороны, не должно быть привязано к конкретной БД.

Диаграмма вариантов использования является исходным концептуальным представлением системы в процессе ее проектирования и разработки. Назначение данной диаграммы заключается в формализации функциональных требований к системе с помощью действующих лиц и вариантов использования. Действующее лицо – потенциальный пользователь системы, в большинстве случаев – человек. Вариант использования – задачи, с которыми действующие лица обращаются к БД.

Составление диаграммы вариантов использований позволяет:

1. Экспертам предметной области описать взгляд на систему извне с такой степенью детализации, что разработчики сумеют сконструировать ее внутреннее представление;
2. Варианты использования позволяют разработчикам понять назначение разрабатываемой БД;
3. Варианты использования являются основой тестирования элементов БД на всем протяжении его жизненного цикла.

Администратор гостиницы может забронировать комнату, ввести журнал оформления услуг клиента, контролировать оплату, регистрировать посетителей, вести учет постельного белья и полотенец, а также управлять наймом сотрудников.

Горничная может делать уборку, а также стирать постельного белье и полотенца.

С проектируемой системой будут взаимодействовать следующие действующие лица:

* Администратор
* Горничная

В реальной жизни посетитель (клиент) гостиницы может на прямую или позвонить на телефон администратору гостиницы и узнать всю интересующую информацию о комнатах или об отеле, а также забронировать комнату.

Однако в проектируемой системе такой возможности у него нет, поэтому посетитель не будет являться действующим лицом.

Рассмотрим варианты использования каждого из действующих лиц.

Администратору доступны следующие функции:

* Управлять бронированием.
* Регистрировать посетителя.
* Вести учет комнат.
* Вести учет постельного белья.
* Управлять наймом сотрудников.
* Составлять отчеты для руководства.
* Контроль оплаты счетов.
* Оформлять услуг и клиента.

Горничной доступны следующие функции:

* Управлять состояние уборки комнаты.
* Вести учет постельного белья.

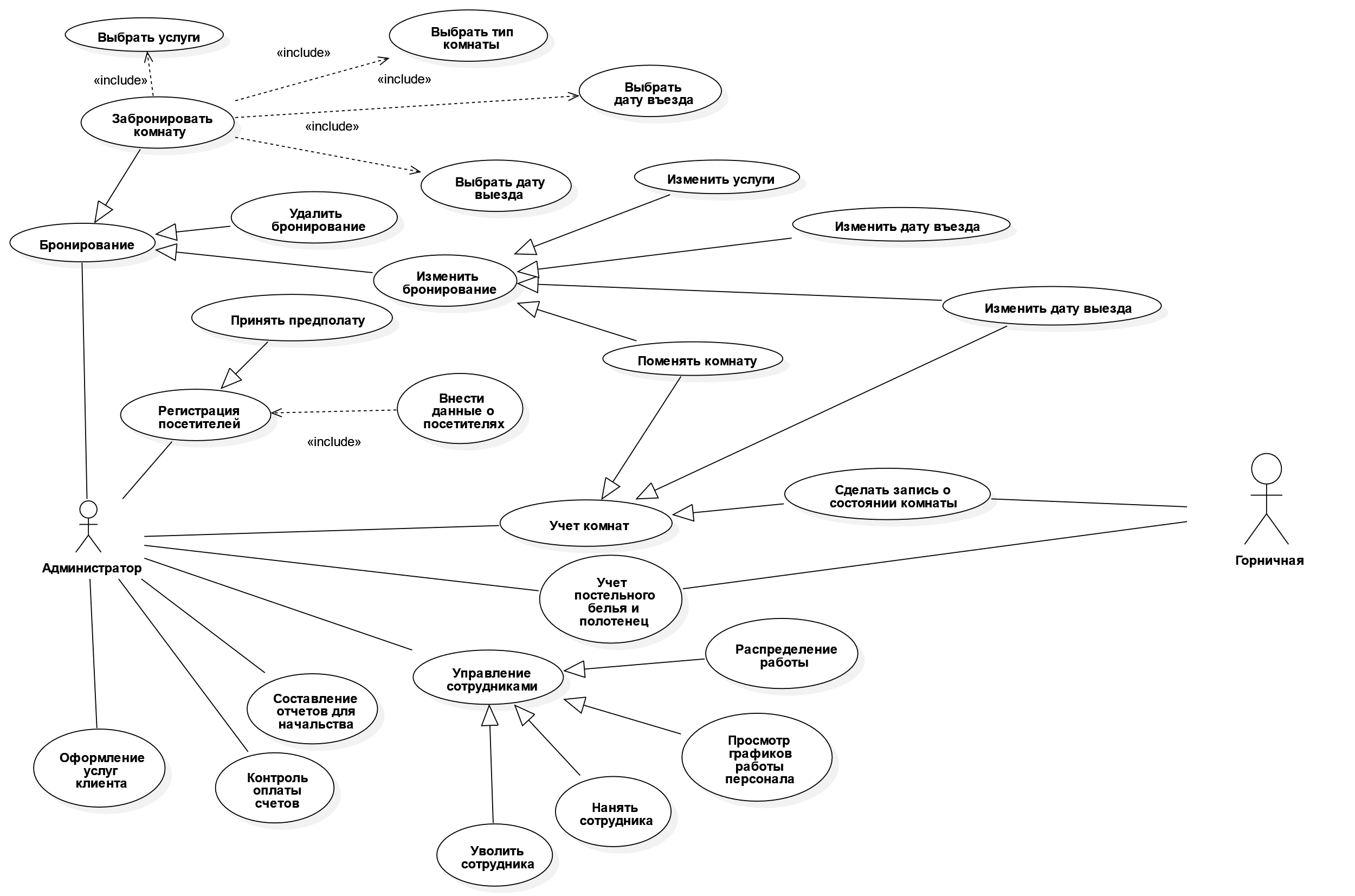
Составим общую диаграмму использования (рис. 2.1.) 

Рис. 2.1 Диаграмма вариантов взаимодействия

В первом приближении для решения выделенных задач необходимо хранение данных о следующих объектах:

* Бронирование – содержит информацию о выбранных услугах, типе номера, дате въезда и выезда, количестве человек.
* Посетитель – содержит ФИО, паспорт (документы подтверждающие личность), дата заселения и номер занятой комнаты.
* Заселение – содержит в себе посетителей, и занятые номера
* Номер – содержит информацию о типе комнаты, уникальный номер, этаж, наличие вида во двор, на море (если есть).
* Сотрудник – содержит информацию о графике работы сотрудника, а также его индивидуальную информацию, такую как ФИО, паспорт, год рождения, тип занятости.
* Работа – содержит информацию о сотруднике, который выполнил данную работу, вид выполненной работы и время завершения работы.
* Услуги – содержат информацию об услуге и её стоимости.
* Тип комнаты – содержит информацию о ее названии, размерах, вместимости, наличии душа и кровати.

ГЛАВА 3. Логическое проектирование

Логическое проектирование реляционной базы данных – создание схемы базы данных на основе конкретной модели данных. Целью логического проектирования является определение состава и структуры таблиц БД на основе результатов концептуального проектирования и проверка полученной модели с помощью методов нормализации.

ER-диаграмма представляет собой логическую модель данных. Назначение данной диаграммы заключается в графическом описании объектов предметной области и связей между ними.

Базис понятий семантического моделирования в общем случае включает:

* определение объекта-экземпляра;
* определение объекта-типа;
* определение связи между объектами;
* определение свойства объекта;
* определение идентифицирующего свойства объекта.

Для ER-диаграммы базовыми являются понятия:

* сущность – это абстрактный объект, который в конкретном контексте имеет независимое существование.
* связь – это ассоциация сущностей или отношение между сущностями;
* атрибут – это свойство сущности или связи.

В прошлой главе были выделены объекты, которые надо хранить в БД. Они станут сущностями при ER-моделировании

Посетитель делает обязательно одно или несколько бронирований, каждое бронирование обязательно бронируется одним посетителем (рис. 3.1).

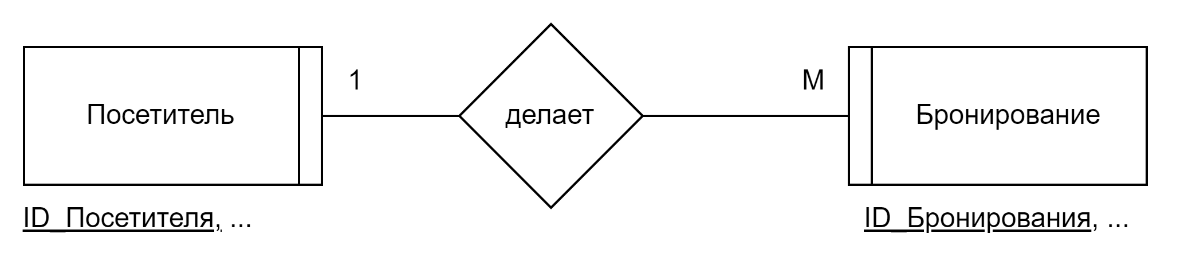


Рис. 3.1. ER-диаграмма связи ДЕЛАЕТ

Бронирование занимает один номер, номер занимается многими или не одним бронированием(рис. 3.2).



Рис. 3.2. ER-диаграмма связи ЗАНИМАЕТ

Заселение содержит один номер, каждый номер содержит хотя бы одно или несколько заселений (рис. 3.3).



Рис. 3.3. ER-диаграмма связи СОДЕРЖИТ

Заселение содержит несколько посетителей, посетитель хотя бы в одном или нескольких заселениях (рис. 3.4).



Рис. 3.4. ER-диаграмма связи СОДЕРЖИТ

Заселение может иметь несколько услуг, услуги могут иметь несколько заселений(рис. 3.5).



Рис. 3.5. ER-диаграмма связи СОДЕРЖИТ

Работа обязательно происходит в одном номере, в номере может быть много работ (рис. 3.6).



Рис. 3.6. ER-диаграмма связи ПРОИСХОДИТ В

Работа выполняется одним сотрудником, сотрудник обязательно выполняет одну или несколько работ (рис. 3.7).



Рис. 3.7. ER-диаграмма связи ВЫПОЛНЯЕТ

Каждый номер обязательно содержит тип номера, каждый тип номера может иметь обязательно один или несколько номеров (рис. 3.8).



Рис. 3.8. ER-диаграмма связи СОДЕРЖИТ

Сформируем набор предварительных отношений с указанием предполагаемого первичного ключа для каждого отношения.

Связь ДЕЛАЕТ удовлетворяет условиям правила 4, в соответствии с которым получаем 2 отношения:

1. Посетитель(ID\_Посетителя, …)
2. Бронирование (ID\_Бронирования, ID\_Посетителя…) – добавился неключевой атрибут ID\_Посетителя

Связь ЗАНИМАЕТ удовлетворяет условия правила 4, в соответствии с которым получаем 2 отношения:

1. Бронирование (ID\_Бронирования, ID\_Номера…) – добавился неключевой атрибут ID\_Номера
2. Номер (ID\_Номера,…)

Связь СОДЕРЖИТ удовлетворяет условиям правила 4, в соответствии с которым получаем 2 отношения:

1. Заселение (ID\_Заселения, ID\_Номера…) – добавился неключевой атрибут ID\_Номера
2. Номер (ID\_Номера,…)

Связь СОДЕРЖИТ удовлетворяет условиям правила 6, в соответствии с которым получаем 3 отношения:

1. Заселение (ID\_Заселения,…)
2. Посетитель (ID\_Посетителя, …)
3. Проживающие (ID\_Заселения, ID\_Посетителя)

Связь СОДЕРЖИТ удовлетворяет условиям правила 6, в соответствии с которым получаем 3 отношения:

1. Заселение (ID\_Заселения,…)
2. Услуга (Название услуги, …)
3. Услуги заселения (ID\_Заселения, Название услуги)

Связь ПРОИСХОДИТ В удовлетворяет условиям правила 4, в соответствии с которым получаем отношения:

1. Работа (ID\_Работы, ID\_Номера,…) – добавился неключевой атрибут ID\_Номера.
2. Номер( ID\_Номера,…)

Связь ПРОИСХОДИТ В удовлетворяет условиям правила 4, в соответствии с которым получаем 2 отношения:

1. Работа (ID\_Работы, ID\_Сотрудника, …) – добавился неключевой атрибут ID\_Сотрудника
2. Сотрудник (ID\_Сотрудника, …)

Связь СОДЕРЖИТ удовлетворяет условиям правила 4, в соответствии с которым получаем 2 отношения:

1. Номер(ID\_Номера, Категория, …) – добавился неключевой атрибут Категория
2. Тип номера(Категория, …)

Добавим неключевые атрибуты в каждое из предварительных отношений с условием, чтобы отношения отвечали требованиям третьей нормальной формы.

Посетитель(ID\_Посетителя, ФИО, Серия, Номер Паспорта, Номер телефона)

Бронирование (ID\_Бронирования, ID\_Номера, ID\_Посетителя, Дата въезда, Дата выезда, Количество человек)

Номер (ID\_Номера, Категория, Состояние, вид из окна)

Тип Номера(Категория, стоимость, вместительность, размер, наличие душа, тип кровати, наличие кондиционера)

Заселение (ID\_Заселения, ID\_Номера, дата заселения, количество оплачиваемых дней)

Проживающие (ID\_Заселения, ID\_Посетителя)

Услуга (Название услуги, Стоимость)

Услуги заселения (ID\_Заселения, Название услуги, Количество)

Работа (ID\_Работы, ID\_Сотрудника, ID\_Номера, Название работы, Время выполнения)

Сотрудник (ID\_Сотрудника, ФИО, Должность, Номер телефона, Дата рождения)

**ГЛАВА 4. Физическое проектирование**

Физическое проектирование базы данных - процесс подготовки описания реализации базы данных. На этом этапе рассматриваются основные отношения, организация файлов и индексов, предназначенных для обеспечения эффективного доступа к данным, а также все связанные с этим ограничения целостности и средства защиты.

Физическое проектирование является третьим и последним этапом создания проекта базы данных, при выполнении которого проектировщик принимает решения о способах реализации разрабатываемой базы данных. Во время предыдущего этапа проектирования была определена логическая структура базы данных (которая описывает отношения и ограничения в рассматриваемой прикладной области). Хотя эта структура не зависит от конкретной целевой СУБД, она создается с учетом выбранной модели хранения данных, например реляционной, сетевой или иерархической. Однако, приступая к физическому проектированию базы данных, прежде всего необходимо выбрать конкретную целевую СУБД. Поэтому физическое проектирование неразрывно связано с конкретной СУБД. Между логическим и физическим проектированием существует постоянная обратная связь, так как решения, принимаемые на этапе физического проектирования с целью повышения производительности системы, способны повлиять на структуру логической модели данных. В данной работе была выбрана СУБД PostgreSQL.

PostgreSQL — это популярная свободная объектно-реляционная система управления базами данных. PostgreSQL базируется на языке SQL и поддерживает многочисленные возможности.

**Преимущества PostgreSQL:**

* поддержка БД неограниченного размера;
* мощные и надёжные механизмы транзакций и репликации;
* расширяемая система встроенных языков программирования и поддержка загрузки C-совместимых модулей;
* наследование;
* легкая расширяемость.

**Текущие ограничения PostgreSQL:**

* Нет ограничений на максимальный размер базы данных
* Нет ограничений на количество записей в таблице
* Нет ограничений на количество индексов в таблице
* Максимальный размер таблицы — 32 Тбайт
* Максимальный размер записи — 1,6 Тбайт
* Максимальный размер поля — 1 Гбайт
* Максимум полей в записи250—1600 (в зависимости от типов полей)

**Функции** в PostgreSQL являются блоками кода, исполняемыми на сервере, а не на клиенте БД. Хотя они могут писаться на чистом SQL, реализация дополнительной логики, например, условных переходов и циклов, выходит за рамки SQL и требует использования некоторых языковых расширений. Функции могут писаться с использованием различных языков программирования. PostgreSQL допускает использование функций, возвращающих набор записей, который далее можно использовать так же, как и результат выполнения обычного запроса. Функции могут выполняться как с правами их создателя, так и с правами текущего пользователя. Иногда функции отождествляются с хранимыми процедурами, однако между этими понятиями есть различие.

**Расширение** PostgreSQL для собственных нужд возможно практически в любом аспекте. Есть возможность добавлять собственные преобразования типов, типы данных, домены (пользовательские типы с изначально наложенными ограничениями), функции (включая агрегатные), индексы, операторы (включая переопределение уже существующих) и процедурные языки.

**Наследование** в PostgreSQL реализовано на уровне таблиц. Таблицы могут наследовать характеристики и наборы полей от других таблиц (родительских). При этом данные, добавленные в порождённую таблицу, автоматически будут участвовать (если это не указано отдельно) в запросах к родительской таблице.

Для отношений, полученных на предыдущем этапе, будут сформированы следующие таблицы (табл. 1-9).

Таблица 1

Требования к таблице Посетитель

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Столбец** | **Тип данных** | **Нуль?** | **Ключ** | **Значение по умолчанию** | **Ограничение** | **Ссылка** |
| Id Посетителя | serial | Нет | Да |  |  |  |
| ФИО | varchar(255) | Нет |  |  |  |  |
| Серия | int | Да |  |  | > 0 |  |
| Номер паспорта | int | Да |  |  | > 0 |  |
| Номер телефона | varchar(20) | Да |  |  |  |  |

Таблица 2

Требования к таблице Тип номера

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Столбец** | **Тип данных** | **Нуль?** | **Ключ** | **Значение по умолчанию** | **Ограничение** | **Ссылка** |
| Категория | varchar(100) | Нет | да |  |  |  |
| Цена | int | Нет |  | 1 | > 0 |  |
| Вместимость | int | Нет |  | 1 | >0 |  |
| Размер | int | Нет |  | 10 | >0 |  |
| Душ | bool | Нет |  | false |  |  |
| Тип кровати | varchar(30) | Нет |  |  |  |  |
| Кондиционер | bool | Нет |  | false |  |  |

Таблица 3

Требования к таблице Номер

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Столбец** | **Тип данных** | **Нуль?** | **Ключ** | **Значение по умолчанию** | **Ссылка** |
| Id Номера | serial | Нет | Да |  |  |
| Категория | varchar(100) | Нет |  |  | Категория в Тип комнаты |
| Статус | varchar(50) | Да |  |  |  |
| Вид из окна | varchar(100) | Нет |  |  |  |

Таблица 4

Требования к таблице Бронирование

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Столбец** | **Тип данных** | **Нуль?** | **Ключ** | **Значение по умолчанию** | **Ограничения** | **Ссылка** |
| Id Бронирования | serial | Нет | Да |  |  |  |
| Id Посетителя | int | Нет |  |  |  | Id Посетителя в Посетитель |
| Id Номера | int | Нет |  |  |  | Id Номера в Номер |
| Дата въезда | date | Нет |  | NOW () |  |  |
| Дата выезда | date | Нет |  |  | >=Дата въезда |  |
| Количество людей | smallint | Нет |  | 1 | > 0 |  |

Таблица 5

Требования к таблице Заселение

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Столбец** | **Тип данных** | **Нуль?** | **Ключ** | **Значение по умолчанию** | **Ограничения** | **Ссылка** |
| Id Заселения | serial | Нет | Да |  |  |  |
| Id Номера | int | Нет |  |  |  | Id Номера в Номер |
| Дата заселения | date | Нет |  | NOW () |  |  |
| Количество оплачиваемых дней | int | Нет |  |  | >= 0 |  |

Таблица 6

Требование к таблице Проживающие

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Столбец** | **Тип данных** | **Нуль?** | **Ключ** | **Значение по умолчанию** | **Ссылка** |
| Id Заселения | int | Нет | Да |  | Id Заселения в Заселение |
| Id Посетителя | int | Нет | Да |  | Id Посетителя в Посетитель |

Таблица 7

Требования к таблице Услуги

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Столбец** | **Тип данных** | **Нуль?** | **Ключ** | **Значение по умолчанию** | **Ограничения** | **Ссылка** |
| Название услуги | varchar(100) | Нет | Да |  |  |  |
| Цена | int | Нет |  |  | > 0 |  |

Таблица 8

Требования к таблице Услуги заселения

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Столбец** | **Тип данных** | **Нуль?** | **Ключ** | **Значение по умолчанию** | **Ограничения** | **Ссылка** |
| Название услуги | varchar(100) | Нет | Да |  |  | Название услуги в Услуги |
| Id Заселения | int | Нет | Да |  |  | Id Заселения в Заселение |
| Количество | int | Нет |  | 1 | >= 1 |  |

Таблица 9

Требования к таблице Сотрудник

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Столбец** | **Тип данных** | **Нуль?** | **Ключ** | **Значение по умолчанию** | **Ссылка** |
| Id Сотрудника | serial | Нет | Да |  |  |
| ФИО | varchar(255) | Нет |  |  |  |
| Должность | varchar(255) | Нет |  |  |  |
| Номер телефона | varchar(20) | Нет |  |  |  |
| День рождения | date | Нет |  |  |  |

Таблица 10

Требование к таблице Работа

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Столбец** | **Тип данных** | **Нуль?** | **Ключ** | **Значение по умолчанию** | **Ссылка** |
| Id Работы | serial | Нет | Да |  |  |
| Id Сотрудника | int | Нет |  |  | Id Сотрудника в Сотрудник |
| Id Номера | int | Нет |  |  | Id Номера в Номер |
| Название | varchar(100) | Нет |  |  |  |
| Время выполнения | date | Нет |  | NOW () |  |

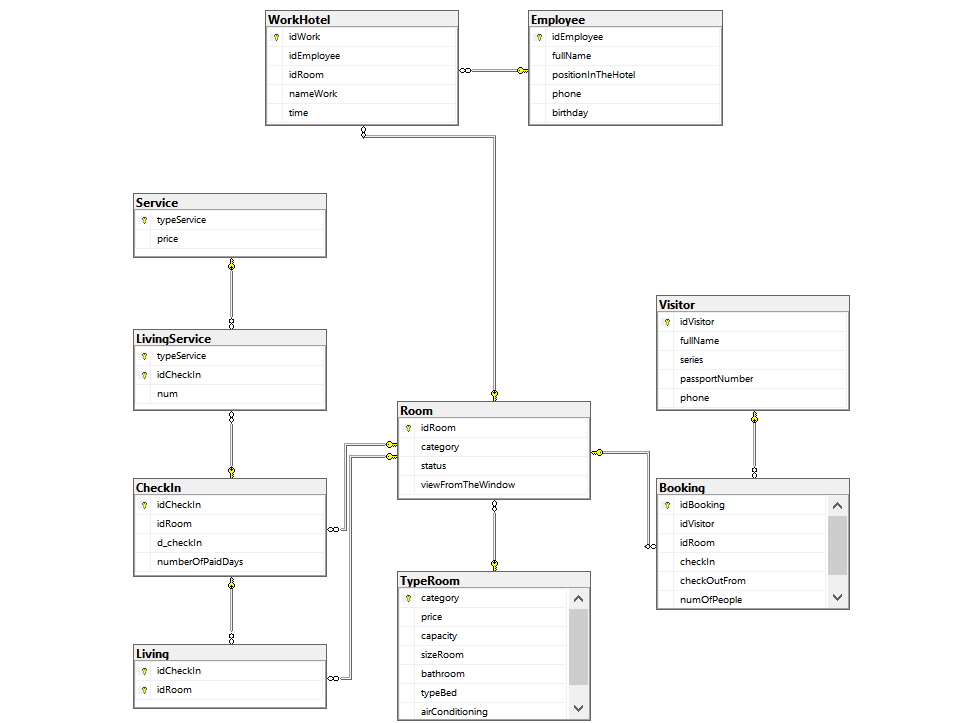
На рисунке 4.1. представлена диаграмма базы данных. 

Рис. 4.1. Диаграмма базы данных

**Заключение**

Для самостоятельной работы была выбрана предметная область «Проектирование БД гостиницы.

Актуальность темы заключается в необходимости автоматизации работы администратора гостиницы с целью повышения уровня информатизации, создания электронного архива, возможности быстрого и надежного способа получения статистики по работе гостиницы.

В ходе работы были получены следующие результаты:

В первой главе был проведен анализ ПО уже существующих программных продуктов, таких как «1С: Предприятие 8. Отель» и к**онфигурация "Простая гостиница"** для выделения нужного функционала. На основе проведенного анализа были определены задачи для выполнения данной работы.

На этапе концептуального проектирования были выделены основные пользователи системы (администратор, горничная), а также были определены функции, составлена диаграмма вариантов использования. Были выявлены объекты, информацию для которых необходимо хранить в БД.

На этапе логического проектирования была разработана логическая модель БД гостиницы, включая в себя ER-диаграммы, на основе которых были сформированы основные отношения (Посетитель, Бронирование, Номер, Заселение, Тип номера, Услуги, Проживающие, Услуги заселения, Сотрудник, Работа), хранящие информацию об объектах гостиницы, а также добавлены неключевые атрибуты.

На этапе физического проектирования была выбрана СУБД PostgreSQL, а также были определены структуры к таблицам. Так же база данных была заполнена тестовыми данными.

**ПРИЛОЖЕНИЕ А**

**SQL-операторы для создания таблиц**

1. Создание таблицы Посетитель.

CREATE TABLE Visitor(

idVisitor SERIAL PRIMARY KEY,

fullName VARCHAR(255) NOT NULL,

series INT CHECK(series > 0),

passportNumber INT CHECK(passportNumber > 0),

phone VARCHAR(20) UNIQUE,

UNIQUE (series, passportNumber)

);

2. Создание таблицы Тип номера.

CREATE TABLE TypeRoom (

category varchar(100) PRIMARY KEY,

price INT NOT NULL DEFAULT 1 CHECK (price > 0),

capacity INT NOT NULL DEFAULT 1 CHECK(capacity > 0),

sizeRoom INT NOT NULL DEFAULT 10 CHECK(sizeRoom > 0),

bathroom BOOL NOT NULL DEFAULT FALSE,

typeBed VARCHAR(30) NOT NULL,

airConditioning BOOL NOT NULL DEFAULT FALSE

);

3. Создание таблицы Номер.

CREATE TABLE Room (

idRoom SERIAL PRIMARY KEY,

category VARCHAR(100) NOT NULL,

status VARCHAR(50),

viewFromTheWindow VARCHAR(100) NOT NULL,

FOREIGN KEY(category)

REFERENCES TypeRoom (category)

);

4. Создание таблицы Бронирование.

CREATE TABLE Booking(

idBooking SERIAL PRIMARY KEY,

idVisitor INT CHECK (idVisitor > 0) NOT NULL,

idRoom INT CHECK(idRoom > 0) NOT NULL,

checkIn DATE NOT NULL DEFAULT NOW(),

checkOutFrom DATE NOT NULL CHECK (checkOutFrom >= checkIn),

numOfPeople SMALLINT DEFAULT 1 CHECK( numOfPeople > 0) NOT NULL,

FOREIGN KEY (idVisitor)

REFERENCES Visitor (idVisitor),

FOREIGN KEY (idRoom)

REFERENCES Room(idRoom)

);

5. Создание таблицы Засление.

CREATE TABLE CheckIn (

idCheckIn SERIAL PRIMARY KEY,

idRoom INT NOT NULL,

d\_checkIn DATE DEFAULT NOW() NOT NULL,

numberOfPaidDays INT NOT NULL CHECK(numberOfPaidDays >= 0),

FOREIGN KEY(idRoom)

REFERENCES Room(idRoom)

);

6. Создание таблицы Проживающие.

CREATE TABLE Living (

idCheckIn INT,

idVisitor INT,

PRIMARY KEY (idCheckIn, idVisitor),

FOREIGN KEY(idCheckIn)

REFERENCES CheckIn(idCheckIn),

FOREIGN KEY(idVisitor)

REFERENCES Visitor (idVisitor)

);

7. Создание таблицы Услуги.

CREATE TABLE Service (

typeService VARCHAR (100) PRIMARY KEY,

price INT NOT NULL CHECK (price > 0)

);

8. Создание таблицы Услуги заселения.

CREATE TABLE LivingService(

typeService VARCHAR (100),

idCheckIn INT,

num INT DEFAULT 1 CHECK(num >=1) NOT NULL,

PRIMARY KEY (typeService, idCheckIn),

FOREIGN KEY (typeService)

REFERENCES Service (typeService),

FOREIGN KEY (idCheckIn)

REFERENCES CheckIn (idCheckIn)

);

9. Создание таблицы Сотрудник.

CREATE TABLE Employee (

idEmployee SERIAL PRIMARY KEY,

fullName VARCHAR (255) NOT NULL,

positionInTheHotel VARCHAR (255) NOT NULL,

phone VARCHAR(20) UNIQUE DEFAULT NULL,

birthday DATE NOT NULL

);

10. Создание таблицы Работа.

CREATE TABLE WorkHotel (

idWork SERIAL PRIMARY KEY,

idEmployee INT NOT NULL,

idRoom INT NOT NULL,

nameWork VARCHAR(100) NOT NULL,

time DATE NOT NULL DEFAULT NOW(),

FOREIGN KEY (idEmployee)

REFERENCES Employee (idEmployee),

FOREIGN KEY (idRoom)

REFERENCES Room (idRoom)

);

**ПРИЛОЖЕНИЕ Б**

**Заполнение данными**

****

Рис. Б.1. Данные таблицы «Посетитель»

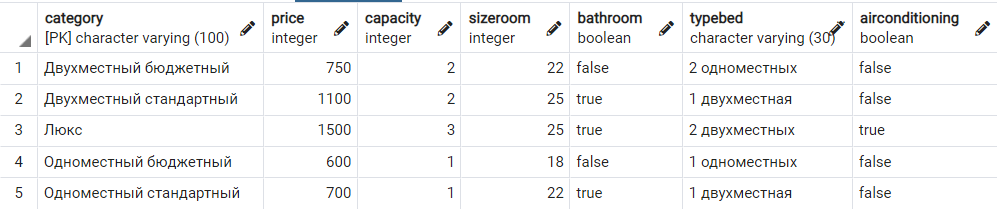


Рис. Б.2. Данные таблицы «Тип номера»

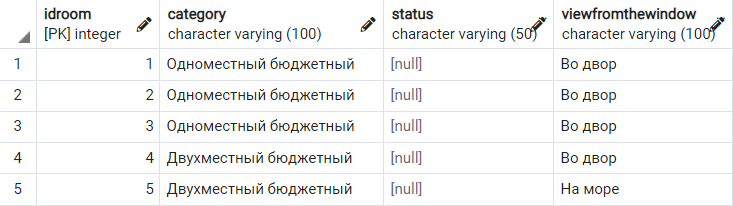
****

Рис. Б.3. Данные таблицы «Номера»

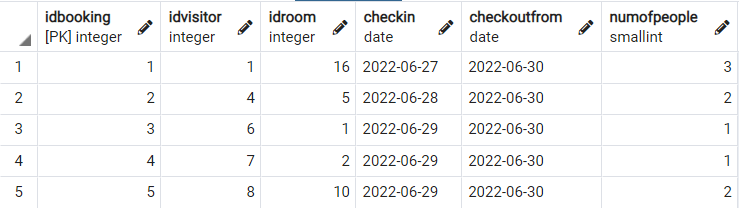
****

Рис. Б.4. Данные таблицы «Бронирование»

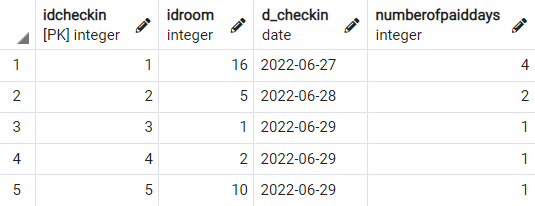
****

Рис. Б.5. Данные таблицы «Заселение»

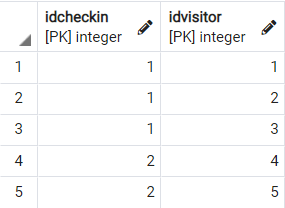
****

Рис. Б.6. Данные таблицы «Проживающие»

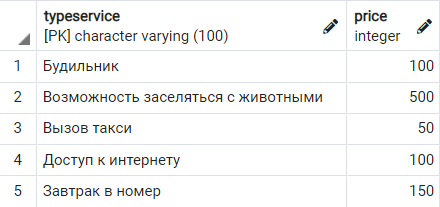
****

Рис. Б.7. Данные таблицы «Услуги»

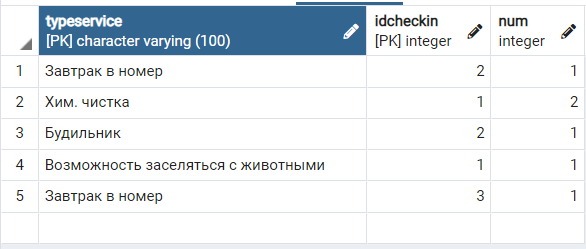
****

Рис. Б.8. Данные таблицы «Услуги проживающих»

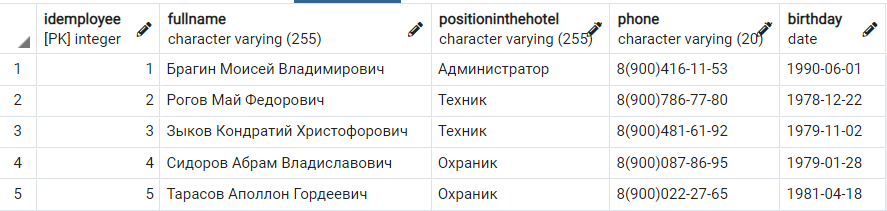


Рис. Б.9. Данные таблицы «Сотрудник»

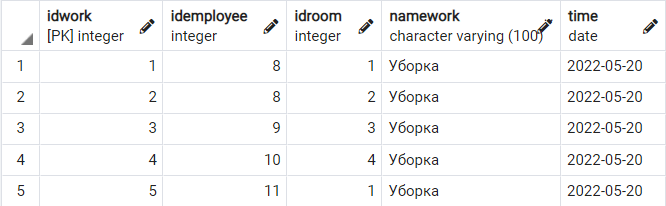
****

Рис. Б.10. Данные таблицы «Работа»