** ОБЛАСТНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«ТОМСКИЙ ЭКОНОМИКО-ПРОМЫШЛЕННЫЙ КОЛЛЕДЖ»**

**ПРОМЕЖУТОЧНАЯ АТТЕСТАЦИЯ**

**по учебной дисциплине**

**«ОП.08 Основы проектирования баз данных»**

*СПЕЦИАЛЬНОСТЬ 09.02.07 ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ПРОГРАММИРОВАНИЕ*

Выполнил:

Студент группы 1922с

Глазунов И.С.

Томск 2024

**ОГЛАВЛЕНИЕ**

[ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 3](#_Toc170077893)

[КОНЦЕПТУАЛЬНАЯ МОДЕЛЬ ДАННЫХ 3](#_Toc170077894)

[НОРМАЛИЗАЦИЯ ДАННЫХ 5](#_Toc170077895)

[ЛОГИЧЕСКАЯ МОДЕЛЬ ДАННЫХ 5](#_Toc170077896)

[ФИЗИЧЕСКАЯ МОДЕЛЬ ДАННЫХ 6](#_Toc170077897)

[СЛОВАРЬ ДАННЫХ 7](#_Toc170077898)

# ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Предметной областью в данном тексте является управление финансами и арендными отношениями в крупном торговом центре. Это включает в себя планирование торговых точек, сбор данных о потенциальных клиентах, показ свободных площадей, оформление договоров аренды и ведение базы данных с информацией о торговых точках, клиентах и периодах аренды.

# КОНЦЕПТУАЛЬНАЯ МОДЕЛЬ ДАННЫХ

Концептуальная модель – это отражение предметной области, для которой

разрабатывается база данных. Не вдаваясь в теорию, отметим, что это некая диаграмма с

принятыми обозначениями элементов. Все объекты, обозначающие вещи, обозначаются в

виде прямоугольника. Атрибуты, характеризующие объект - в виде овала, а связи между

объектами - ромбами. Мощность связи обозначаются стрелками (в направлении, где

мощность равна многим - двойная стрелка, а со стороны, где она равна единице - одинарная).

Анализируя предметную область, можно выделить следующие сущности с их атрибутами (см. рисунок 1):

Сущность «Торговые точки» с атрибутами: «Этаж», «Площадь», «Наличие кондиционера» и «Стоимость аренды»

Сущность «Клиенты» с атрибутами: «Название компании», «Адрес», «Телефон», «Реквизиты» и «Контактное лицо»

Сущность «Договоры» с атрибутами: «Идентификатор торговой точки», «Идентификатор клиента», «Начало срока аренды» и «Конец срока аренды»

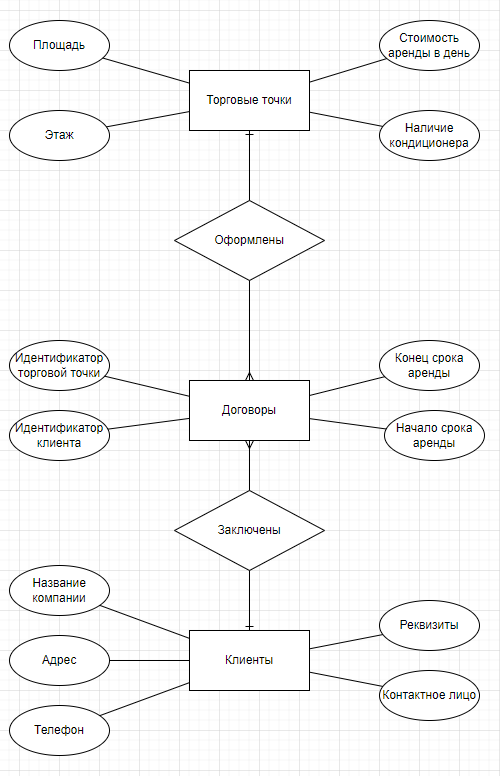


Рисунок 1- Концептуальная модель данных

# НОРМАЛИЗАЦИЯ ДАННЫХ

Нормальная форма — свойство отношения в реляционной модели данных, характеризующее его с точки зрения избыточности, потенциально приводящей к логически ошибочным результатам выборки или изменения данных. Нормальная форма определяется как совокупность требований, которым должно удовлетворять отношение.

Процесс преобразования отношений базы данных к виду, отвечающему нормальным формам, называется нормализацией. Нормализация предназначена для приведения структуры БД к виду, обеспечивающему минимальную логическую избыточность, и не имеет целью уменьшение или увеличение производительности работы или же уменьшение или увеличение физического объёма базы данных. Конечной целью нормализации является уменьшение потенциальной противоречивости, хранимой в базе данных информации.

Устранение избыточности производится, как правило, за счёт декомпозиции отношений таким образом, чтобы в каждом отношении хранились только первичные факты (то есть факты, не выводимые из других хранимых фактов).

Первая нормальная форма (1NF)

Таблицы уже находятся в первой нормальной форме, так как все атрибуты являются атомарными (неделимыми), и каждая таблица имеет первичный ключ.

Вторая нормальная форма (2NF)

Таблицы уже находятся во второй нормальной форме, так как все атрибуты полностью зависят от первичного ключа.

Третья нормальная форма (3NF)

Таблицы уже находятся в третьей нормальной форме, так как нет транзитивных зависимостей между не ключевыми атрибутами.

При проектировании БД третья нормальная форма схем отношений достаточна в. большинстве случаев, и приведением к третьей нормальной форме процесс проектирования реляционной базы данных обычно заканчивается. Однако иногда полезно продолжить процесс нормализации.

# ЛОГИЧЕСКАЯ МОДЕЛЬ ДАННЫХ

Логическая модель базы данных - схема базы данных, выраженная в понятиях модели данных. Этим отличается от концептуальной модели, описывающей семантику предметной области без указания технологии (конкретных методов реализации), и от физической модели, которая описывает конкретные физические механизмы, применяемые для хранения данных в накопителях.

Используя нашу концептуальную модель данных и выполнив нормализацию данных до 3 нормальной формы включительно, построим логическую модель данных, в которой мы можем выделить поля, являющиеся первичными и внешними ключами, необходимые для построения связей.

Получившиеся логическая модель данных представлена на рисунке 2.

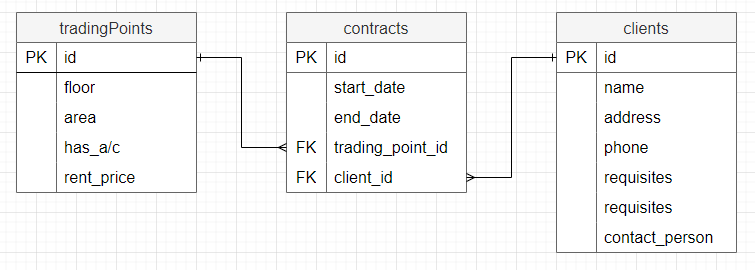


Рисунок 2 - Логическая модель данных

# ФИЗИЧЕСКАЯ МОДЕЛЬ ДАННЫХ

Физическая модель данных – это модель данных, описанная с помощью средств конкретной системы управления базами данных (СУБД). Физическая модель данных строится на даталогической базе путем добавления особенностей конкретной СУБД. К таким особенностям могут относиться поддерживаемые СУБД типы данных, соглашения о присвоении имен таблицам, атрибутам и т.д. Физическая модель данных фактически является готовым заданием на создание БД, имея которое можно реализовать БД в выбранной СУБД.

На рисунке 3 представленная физическая модель данных для информационной системы «Управление финансами и арендными отношениями».

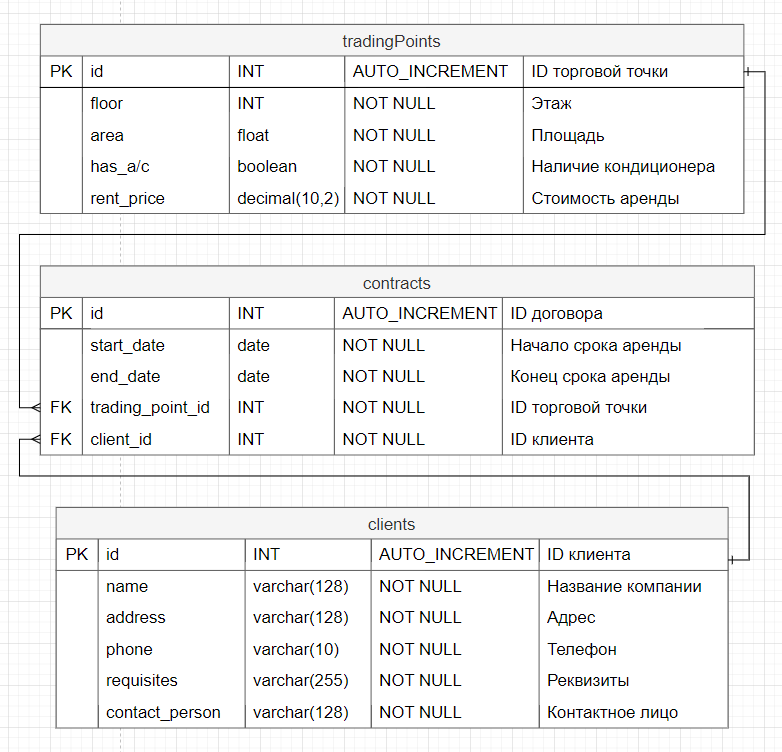


Рисунок 3 - Физическая модель данных

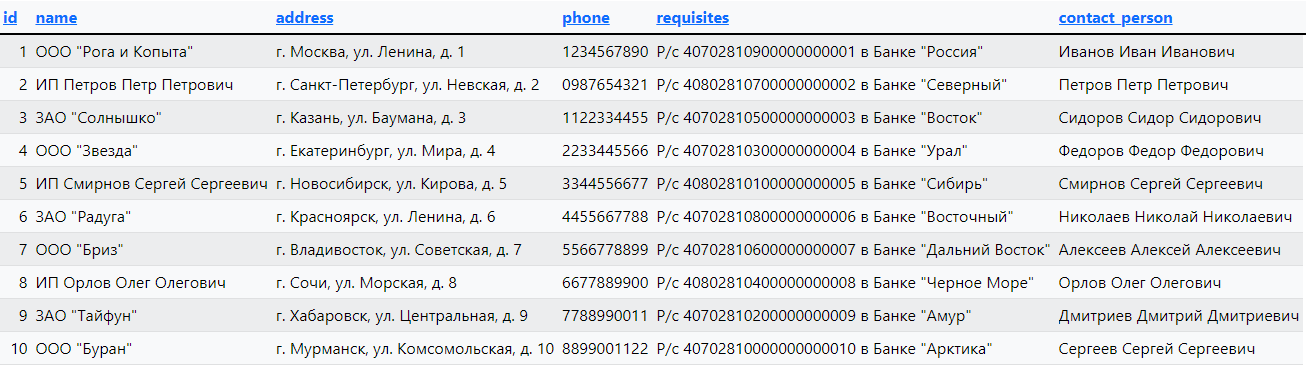
# СЛОВАРЬ ДАННЫХ

Таблица 1 – Словарь данных для БД «Управление финансами и арендными отношениями»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| tradingPoints | | | | |
| Поле | Формат | Обязательное | Ключ | Примечание |
| id | INT | Да | Первичный | ID торговой точки |
| floor | INT | Да |  | Этаж |
| area | float | Да |  | Площадь |
| has\_a/c | boolean | Да |  | Наличие кондиционера |
| rent\_price | decimal(10,2) | Да |  | Цена аренды |
| contracts | | | | |
| Поле | Формат | Обязательное | Ключ | Примечание |
| id | INT | Да | Первичный | ID договора |
| start\_date | date | Да |  | Начало срока аренды |
| end\_date | date | Да |  | Конец срока аренды |
| traiding\_point\_id | INT | Да | Вторичный | Реквизиты |
| client\_id | INT | Да | Вторичный | Контактное лицо |
| clients | | | | |
| Поле | Формат | Обязательное | Ключ | Примечание |
| id | INT | Да | Первичный | ID Клиента |
| name | varchar(128) | Да |  | Название компании |
| address | varchar(128) | Да |  | Адрес |
| phone | varchar(10) | Да |  | Телефон |
| requisites | varchar(255) | Да |  | Реквизиты |
| contact\_person | varchar(128) | Да |  | Контактное лицо |

# ПРИМЕР ТЕКСТОВЫХ ДАННЫХ ДЛЯ КАЖДОГО ОТНОШЕНИЯОГО ОТНОШЕНИЯ

***Таблица clients***



***Таблица contracts***



***Таблица TradingPoints***

******

# SQL-ЗАПРОСЫ И ИХ ОПИСАНИЕ

**1. Запросы с использованием простого оператора выборки данных SELECT и предложения WHERE:**

a. Выведите все данные о клиентах.

SELECT \* FROM Clients;

b. Выведите все данные о клиентах, проживающих в городе Москва.

SELECT \* FROM Clients WHERE address LIKE '%Москва%';

**2. Запросы с использование предикатов AND, OR или NOT:**

a. Выведите все данные о торговых точках, расположенных на первом этаже с кондиционером.

SELECT \* FROM TradingPoints WHERE floor = 1 AND has\_air\_conditioning = TRUE;

b. Выведите все данные о торговых точках, площадь которых больше 70 кв.м. или стоимость аренды в день меньше 1000 рублей.

SELECT \* FROM TradingPoints WHERE area > 70 OR rent\_price\_per\_day < 1000;

**3. Запросы с использование предикатов IN и BETWEEN:**

a. Выведите все данные о клиентах с идентификаторами 1, 3 и 5.

SELECT \* FROM Clients WHERE id IN (1, 3, 5);

b. Выведите все данные о контрактах, начинающихся в период с 1 января 2023 года по 31 декабря 2023 года.

SELECT \* FROM Contracts WHERE start\_date BETWEEN '2023-01-01' AND '2023-12-31';

**4. Запросы с использование явных операций соединения двух и более таблиц:**

a. Выведите идентификатор контракта, дату начала и окончания, а также этаж и площадь торговой точки.

SELECT c.id, c.start\_date, c.end\_date, tp.floor, tp.area

FROM Contracts c

INNER JOIN TradingPoints tp ON c.trading\_point\_id = tp.id;

b. Выведите идентификатор контракта, дату начала и окончания, а также название клиента.

SELECT c.id, c.start\_date, c.end\_date, cl.name AS client\_name

FROM Contracts c

INNER JOIN Clients cl ON c.client\_id = cl.id;

**5. Запросы с использование объединения (UNION):**

a. Выведите названия всех торговых точек и клиентов, в названии которых есть "ООО".

SELECT name FROM TradingPoints WHERE name LIKE '%ООО%'

UNION

SELECT name FROM Clients WHERE name LIKE '%ООО%';

b. Выведите названия всех торговых точек и клиентов, в названии которых есть "Магазин" или "ИП".

SELECT name FROM TradingPoints WHERE name LIKE '%Магазин%'

UNION

SELECT name FROM Clients WHERE name LIKE '%ИП%';

**6. Запросы на получение итоговых значений**

a. Подсчитайте общее количество контрактов.

SELECT COUNT(\*) FROM Contracts;

b. Выведите минимальную и максимальную стоимость аренды в день для торговых точек.

SELECT MIN(rent\_price\_per\_day) AS min\_rent, MAX(rent\_price\_per\_day) AS max\_rent FROM TradingPoints;

**7. Запросы с использование подзапросов (вложенных запросов):**

a. Выведите все данные о торговых точках, которые арендуются клиентом с идентификатором 1.

SELECT \* FROM TradingPoints WHERE id IN (SELECT trading\_point\_id FROM Contracts WHERE client\_id = 1);

b. Выведите все данные о клиентах, которые арендуют торговую точку с идентификатором 2.

SELECT \* FROM Clients WHERE id IN (SELECT client\_id FROM Contracts WHERE trading\_point\_id = 2);

**8. Запросы с использование оператора группировки (GROUP BY):**

a. Подсчитайте количество контрактов для каждого клиента.

SELECT client\_id, COUNT(\*) AS contract\_count FROM Contracts GROUP BY client\_id;

b. Подсчитайте количество контрактов для каждой торговой точки.

SELECT trading\_point\_id, COUNT(\*) AS contract\_count FROM Contracts GROUP BY trading\_point\_id;

**9.** **Запросы с использование предложения HAVING:**

a. Выведите идентификаторы клиентов, у которых более одного контракта.

SELECT client\_id, COUNT(\*) AS contract\_count FROM Contracts GROUP BY client\_id HAVING COUNT(\*) > 1;

b. Выведите идентификаторы торговых точек, у которых более одного контракта.

SELECT trading\_point\_id, COUNT(\*) AS contract\_count FROM Contracts GROUP BY trading\_point\_id HAVING COUNT(\*) > 1;

**10. Запросы на обновление данных (оператор UPDATE):**

a. Обновите адрес клиента с идентификатором 1 на "г. Москва, ул. Новый Арбат, д. 10".

UPDATE Clients SET address = 'г. Москва, ул. Новый Арбат, д. 10' WHERE id = 1;

b. Обновите стоимость аренды в день для торговой точки с идентификатором 1 на 1100 рублей.

UPDATE TradingPoints SET rent\_price\_per\_day = 1100 WHERE id = 1;