**Лабораторная работа №6**

**Инвариантная часть:**

Определите потенциальные ключи для каждого из отношений. Если для каких-то отношений таких ключей окажется несколько, выберите один из них на роль первичного (PK), а остальные пометьте альтернативными (AK).

Определите, какие из атрибутов отношений являются обязательными, т.е. не допускают наличие неизвестных значений. Пометьте такие атрибуты как NOT NULL. (*Задание 1 и 2*)

**Таблички:**

Клиенты:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ИД клиента  PK NOT NULL UQ | Имя  NOT NULL | Фaмилия | Дата рождения | Телефон | Почта  NOT NULL | адрес |

Заказы:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ИД заказа  PK NOT NULL UQ AI | Способ получения  NOT NULL | Дата заказа  NOT NULL | Клиент  FK NOT NULL | Сотрудник  FK NOT NULL | Кол-во игр  NOT NULL | Название игры  FK NOT NULL | Производитель  FK NOT NULL |

Игры:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название игры  PK NOT NULL | Производитель  PK NOT NULL | Мин.кол-во игроков  NOT NULL | Макс.кол-во игроков  NOT NULL | Минимальный возраст |

Сотрудники:

|  |  |  |
| --- | --- | --- |
| ИД сотрудника  PK NOT NULL UQ | Имя  NOT NULL | Фамилия  NOT NULL |

* Сущность “CLients”:

id\_client LIKE 'K%'

e-mail LIKE '%@%.%'

phone\_number LIKE '+7%' OR phone\_number LIKE '8%'

* Сущность “Orders”:

received = 'Самовывоз' OR received = 'Доставка'

client LIKE 'K%'

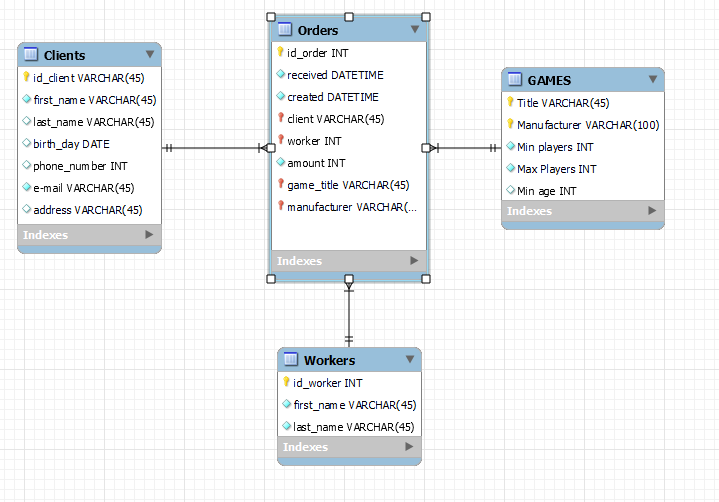
worker LIKE 'C\_\_'

amount >= 1

* Сущность “Workers”:

Min\_players >= 1

*Скрин схемы:*



Связи:

- каждый заказ может содержать в себе несколько позиций (связь: Orders -> GAMES);

- каждая игра может являться позицией многих заказов (связь: GAMES -> Orders).

- один клиент может сделать любое количество заказов (связь: Clients -> Orders);

- один сотрудник может вести несколько заказов (связь: Workers -> Orders);

Запросы:

*Запрос создания: [ссылка](<script src=\"https:/gist.github.com/tarassiky/cb46e05ce8433fc54903903e3328d5b7.js\"></script>)*

*Запрос наполнения таблиц: [ссылка](<script src=\"https:/gist.github.com/tarassiky/f657bd14f6eaa9aac1b1ee6518e0a2ab.js\"></script>)*

**Ответ на вопрос:** Если каждый сотрудник может обработать только один заказ (после чего его увольняют), то связь между таблицами ЗАКАЗ и СОТРУДНИК становится “один-к-одному”.

Что нужно сделать для поддержки этой связи?

Необходимо обеспечить, чтобы атрибут “сотрудник” (worker) в таблице “Заказы” (Orders) был помечен как уникальный (UNIQUE). Это предотвратит назначение одного сотрудника на несколько заказов.

Что может помешать указать одного сотрудника на несколько заказов?

Если попытаться вставить в таблицу “Заказы” несколько записей с одинаковым значением в поле “сотрудник”, база данных выдаст ошибку из-за нарушения ограничения уникальности. Это гарантирует, что каждый заказ будет обработан только одним сотрудником, и каждый сотрудник получит только один заказ.

**Вариативная часть:**

Вариант 1. Проанализировать информацию по избыточности баз данных и привести примеры неверного логического проектирования. Предложить алгоритм решения (по устранению) недостатков логической структуры.

*Результат*

Избыточность — это повторяющееся хранение одних и тех же данных, что может вызвать:

- Проблемы при обновлении (необходимость изменять данные в нескольких местах одновременно)

- Увеличение объема базы данных

- Нарушение целостности информации

*Примеры избыточности:*

- В таблице "Заказы" могут дублироваться сведения о клиентах (ФИО, адрес доставки)

- Стоимость игры может храниться и в таблице "Игры", и в "Истории заказов"

- Контакты сотрудников могут повторяться в разных записях заказов

*Ошибки проектирования:*

- Нарушение нормальных форм:

- В таблице "Игры" несколько жанров указаны в одной ячейке

- Данные о разработчике повторяются для разных игр

- Цена игры зависит не только от её идентификатора

- Отсутствие целостности данных:

- Нет связей (FOREIGN KEY) между таблицами заказов, клиентов и сотрудников

- Не заданы ограничения на допустимое количество игроков

- Нет проверки корректности email и номеров телефонов

*Методика устранения недостатков*

Шаг 1: Проверить соответствие нормальным формам

- 1НФ: Все поля должны содержать только неделимые значения (без списков)

- 2НФ: Неключевые атрибуты должны зависеть от всего первичного ключа

- 3НФ: Исключить транзитивные зависимости (данные должны зависеть только от PK)

Шаг 2: Устранить дублирование

- Убрать повторяющиеся данные (например, удалить client\_name из таблицы заказов)

- Заменить вычисляемые поля на представления (VIEW) или автоматические триггеры

Шаг 3: Оптимизировать связи между таблицами

- Для связи 1:1 проверить, не стоит ли изменить её на 1:N

- Для связи M:N убедиться в наличии связующей таблицы

Шаг 4: Обеспечить целостность данных

- Добавить PRIMARY KEY для всех основных таблиц

- Настроить FOREIGN KEY для связанных данных

- Ввести ограничения (CHECK) на допустимые значения (например, min\_players > 0)

Шаг 5: Тестирование

Заполнить базу тестовыми данными и проверить:

- Отсутствие дублирования

- Сохранение целостности при изменении и удалении записей