УДК 004.652

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА БАЗЫ ДАННЫХ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Тарарыев М.С.[1], студент гр. 12002208, III курс, Устинов Е.Д.[2], студент гр. ИНБб-211, IV курс, Дунаева В.А.[1], магистрант гр. 12002340, II курс
Научный руководитель: Игрунова С.В. [1], к.с.н., доцент
[1]Федеральное государственное автономное образовательное учреждение высшего образования «Белгородский государственный национальный исследовательский университет»
г. Белгород
[2]Автономная некоммерческая организация высшего образования «Белгородский университет кооперации, экономики и права»
г. Белгород

Проектирование и разработка базы данных информационной системы – это процесс создания структуры хранения данных, который обеспечивает эффективное хранение, управление и доступ к данным. Этот процесс включает несколько этапов, начиная от анализа требований до реализации и тестирования.

В качестве элементов научной новизны реализованных в настоящем исследовании можно выделить разработку и апробацию алгоритма проектирования и разработки базы данных применительно к конкретной информационной системе – отделу кадров коммерческого субъекта, содержащего пошаговый подход.

Пошаговый подход к проектированию и разработке базы данных включает ряд этапов.

На первом этапе осуществляется анализ требований. В процессе важно определить, какая информация будет храниться в базе данных, кто будет ими пользоваться и каким образом эти данные будут обрабатываться. Может так же проводиться анализ бизнес-процессов, ключевых сущностей, атрибутов и связей между ними.

На втором этапе чаще всего реализуют концептуальное проектирование. Концептуальная модель описывает сущности и отношения между ними на высоком уровне абстракции. Она помогает визуализировать структуру будущей базы данных и обеспечить согласованность всех участников проекта. Для концептуального проектирования часто используется нотация UML или ER-диаграммы (Entity Relationship Diagram).

Третий этап подразумевает проектирование логической модели, которая уточняет концептуальную модель, переводя её в термины конкретной системы управления базами данных. В этот этап входит определение типов данных, первичных ключей, внешних ключей и индексов. Часто используются реляционные модели, такие как SQL Server, MySQL, PostgreSQL и другие.

Четвертый этап включает проектирование физической модели, определяющей физическую реализацию базы данных на сервере. Включает выбор типа хранения данных, распределение файлов базы данных, настройки производительности и безопасности. Важно учитывать такие аспекты, как объём данных, частота запросов, нагрузка на сервер и требования к отказоустойчивости.

После завершения проектирования начинается реализация базы данных. Это включает создание таблиц, индексов, триггеров, хранимых процедур и представлений. Чаще всего для этого используют языки программирования, такие как SQL.

Этап тестирования и оптимизации базы данных является важнейшим в описываемом алгоритме действий. Так, тестирование базы данных включает проверку корректности работы запросов, обеспечение целостности данных и тестирование производительности. Целесообразно убедиться, что база данных работает правильно при различных сценариях использования. Если возникают проблемы с производительностью, нужно провести оптимизацию запросов и индексацию.

Заключительным этапом процесса проектирования и разработки базы данных информационной системы выступает ее поддержка и обновление. Это может включать резервное копирование данных, мониторинг производительности, исправление ошибок и внедрение новых функций.

Практическая апробация описанного алгоритма при проектировании и разработке базы данных может быть раскрыто на примере конкретной информационной системы – отдела кадров.

Каждое предприятие имеет в своей структуре отдел кадров – структурное подразделение, которое взаимодействует непосредственно с персоналом. Поэтому важно принять тот факт, что центральным объектом взаимодействия различных частей отдела является сотрудник и информация о нем. Любое предприятие хранит информацию о сотрудниках, отделах и должностях. Происходит постоянное перемещение и изменения состава персонала. Всю эту информацию необходимо где-то хранить [3]. Традиционным способом хранения и ведения такой информации являются бумажные носители. Данным способом пользуются малые и средние предприятия, ввиду простоты и дешевизны данного способа, однако со временем данный способ не обеспечивает должный уровень реагирования и является затратным как по времени так и по денежным средствам при выполнении рутинных задач документооборота, а также есть вероятность появления ошибок [2].

Для решения этих проблем предприятия переходят на электронный документооборот – от простых картотек на файлах типа .docx вплоть до продвинутых систем электронного документооборота (ЭДО) как «amoCRM» или «1С документооборот». Такие системы позволяют оптимизировать бизнес процессы внутри отдела кадров, однако основным недостатком для малых и средних предприятий является экономическая целесообразность в виде низкой стоимости.

Разрабатываемый нами проект достаточно прост в использовании и не требует дополнительных как материальных, так и аппаратных затрат.

В целях проектирования базы данных необходимо определить, какими информационными объектами оперирует сотрудник отдела кадров.

В связи с тем, что основные функциональные обязанности сотрудников отдела кадров включают:

- ведение учета состава персонала;
- ведение учета прохождения различного рода инструктажей;
- ведение внутрифирменного движения персонала.

Различные данные о сотруднике (такие как отдел, в котором он работает или должность которую он занимает) целесообразно отделить как отдельные компоненты для облегчения понимания предметной области и как следствие – упрощение разработки программного интерфейса.

На основе этих данных можно сформировать инфологическую модель базы данных разрабатываемой информационной системой, представленную на рисунке 1[4].

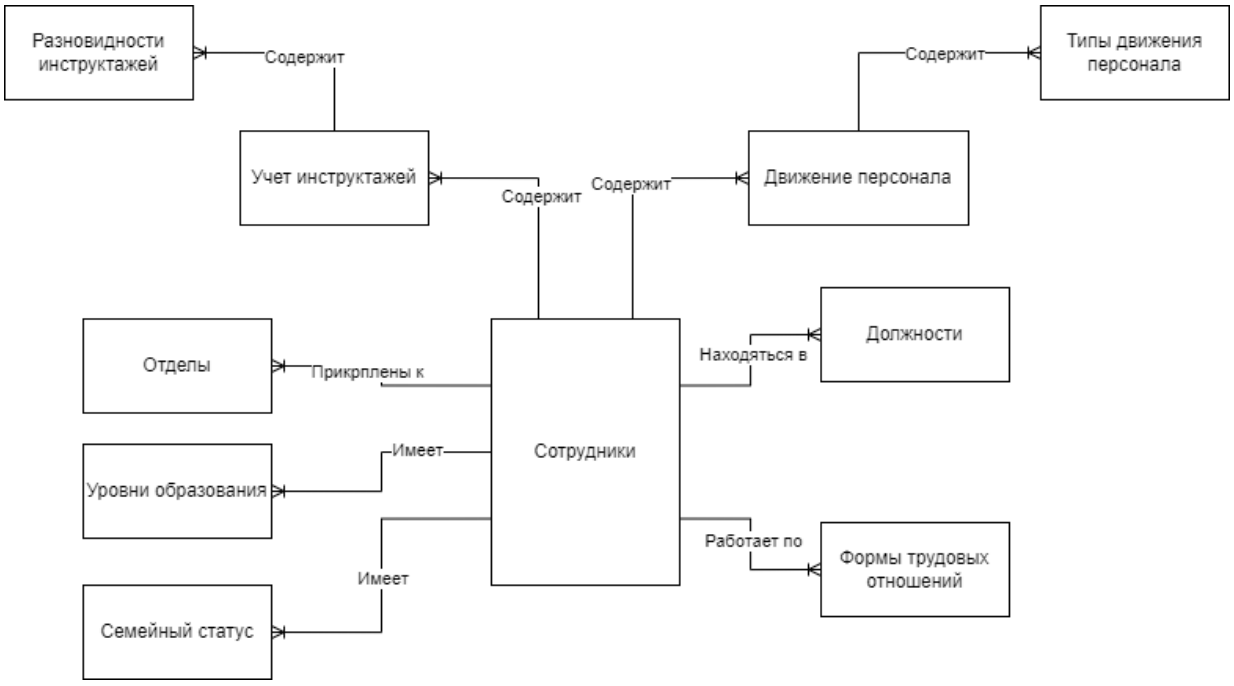

Рисунок 1 – Инфологическая (концептуальная) модель БД

На основании инфологической модели, созданной в предшествующем вопросе мы можем приступить к разработке логической модели базы данных.

В будущей информационной системе нам необходимо создать понятный и простой пользовательский интерфейс, позволяющий быстро и оперативно добавлять, изменять и удалять данные, для этого необходимо как минимум выделить повторяющиеся свойства в отдельные компоненты – информационные (вспомогательные) таблицы, состоящие из кода-идентификатора свойства, его названия и необязательного описания, поясняющее данное свойство.

На основании этих данных мы можем создать логическую модель базы данных, представленную на рисунке 2[4].

Данная модель состоит из трех основных таблиц и семь вспомогательных (информационных) таблиц и вместе образуют структуру разрабатываемой базы данных информационной системы.

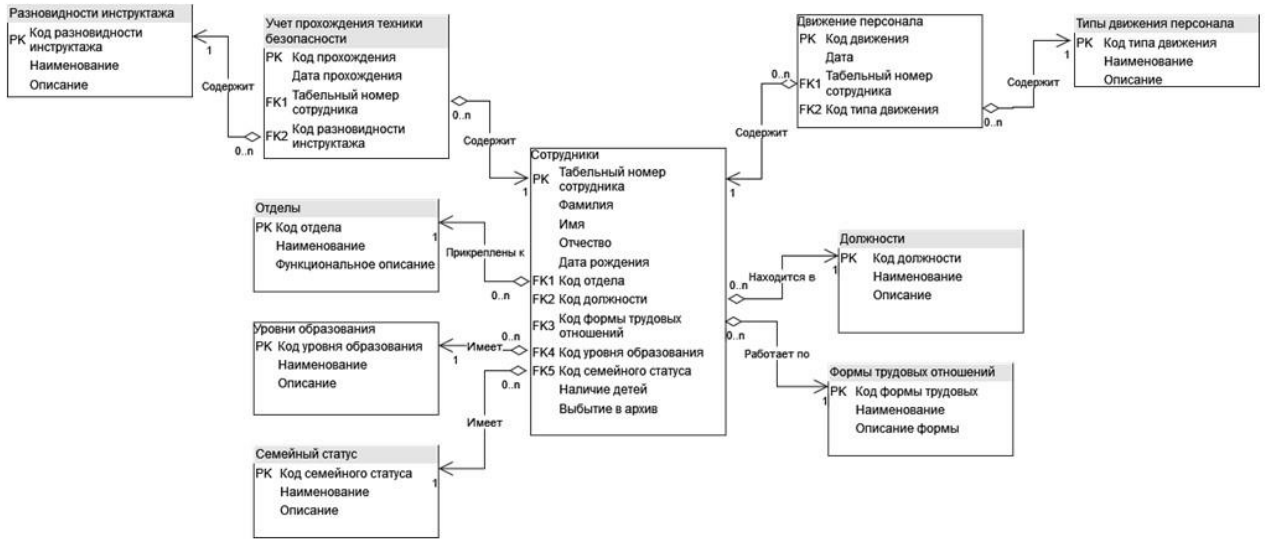

Рисунок 2 – Логическая модель БД

Следующим этапом исследования является создание физической модели базы данных. Для этого необходимо выбрать систему управления базой данных (СУБД).

В современном мире существует много различных СУБД, такие как: PostgreSQL, MySQL, MSSQL, SQLite, Oracle, Firebird, mSQL, Sybase ASE и др.

Так как приложение будет запускаться преимущественно на персональных компьютерах под управлением операционной системой Windows целесообразно использовать СУБД «MSSQL (SQL Server manager)» ввиду простоты использования и понятного интерфейса.

Согласно логической модели БД была создана физическая модель БД, представленная на рисунке 3[4].

По итогу выполнения SQL запросов, представленных на рисунке 4 и 5, БД была создана и поделена на 10 таблиц, 3 из которых являются основными. Описание каждой основной таблицы представлена ниже:

employees (сотрудники) – таблица, представляющее из себя картотеку данных о персонале предприятия, согласно логической модели БД, некоторые поля являются внешним ключем на соответствующие записи в вспомагательных таблицах

staff_movements (передвижения персонала) – таблица, хранящая в себе историю перемещений персонала, состоит из 5 полей (идентификатор, дата передвижения, табельный номер (идентификатор) сотрудника, тип движения (внешний ключ по идентификатору соответствующий внешней таблицы), пояснение (необязательно))

safety_training_records (учет прохождения инструктажей) – таблица записей о прохождении различных инструктажей, также как и таблица staff_movements состоит из 5 аналогичных полей.

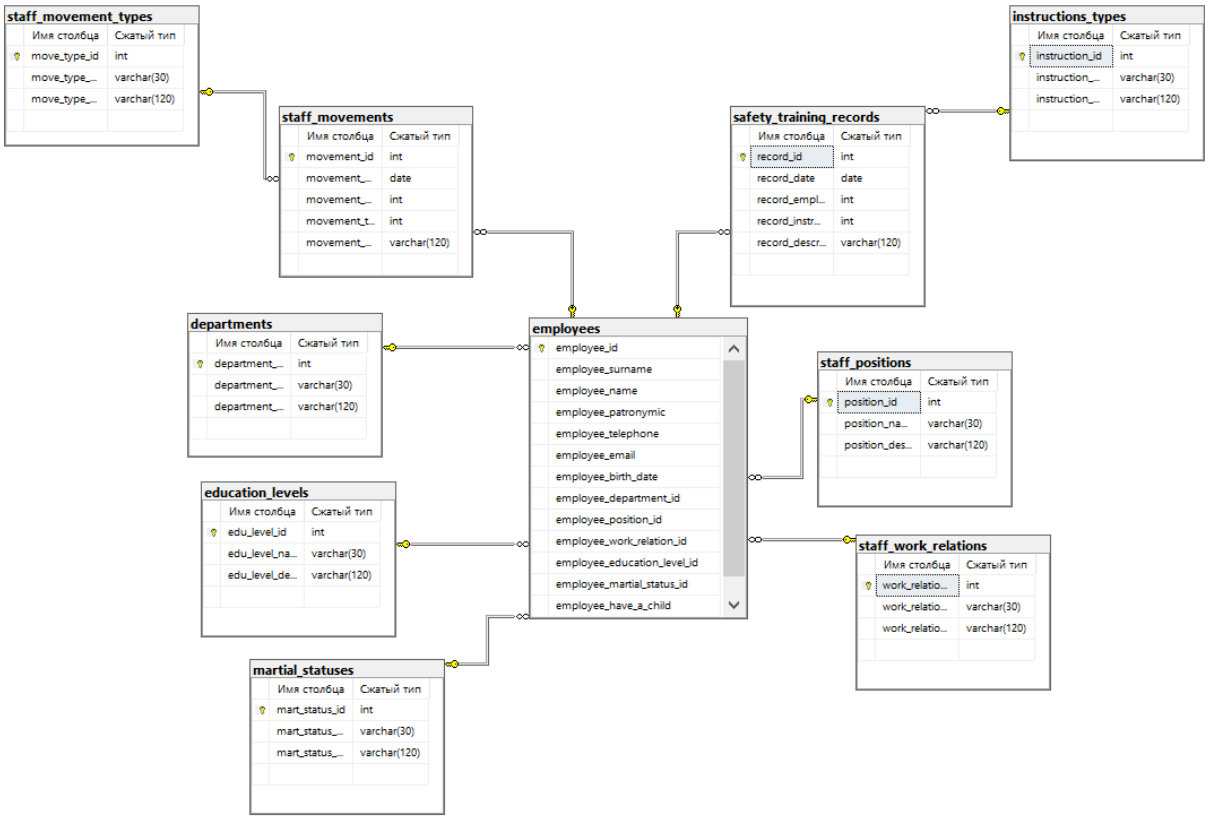

Рисунок 3 – Физическая модель БД

Структура таблиц staff_movements и safety_training_records намеренно совпадают для облегчения взаимодействия с ними.

```
USE HRDepartment;
CREATE TABLE employees(
    employee_id             INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    employee_surname        VARCHAR(30) NOT NULL,
    employee_name           VARCHAR(30) NOT NULL,
    employee_patronymic     VARCHAR(30) NOT NULL,
    employee_birth_date     DATE NOT NULL DEFAULT '01-01-2000',
    employee_department_id  INT NOT NULL
                            FOREIGN KEY REFERENCES dbo.departments(department_id)
                            ON DELETE CASCADE
                            ON UPDATE CASCADE,
    employee_position_id    INT NOT NULL
                            FOREIGN KEY REFERENCES dbo.staff_positions(position_id)
                            ON DELETE CASCADE
                            ON UPDATE CASCADE,
    employee_work_relation_id INT NOT NULL
                            FOREIGN KEY REFERENCES dbo.staff_work_relations(work_relation_id)
                            ON DELETE CASCADE
                            ON UPDATE CASCADE,
    employee_education_level_id INT NOT NULL
                            FOREIGN KEY REFERENCES dbo.education_levels(edu_level_id)
                            ON DELETE CASCADE
                            ON UPDATE CASCADE,
    employee_martial_status_id INT NOT NULL
                            FOREIGN KEY REFERENCES dbo.martial_statuses(mart_status_id)
                            ON DELETE CASCADE
                            ON UPDATE CASCADE,
    employee_have_a_child   BIT NOT NULL DEFAULT 0,
    employee_in_archive     BIT NOT NULL DEFAULT 0
);
CREATE TABLE safety_training_records(
    record_id               INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    record_date             DATE NOT NULL DEFAULT GETDATE(),
    record_employee         INT NOT NULL FOREIGN KEY REFERENCES dbo.employees(employee_id)
                            ON DELETE CASCADE
                            ON UPDATE CASCADE,
    record_instruction_type_id INT NOT NULL FOREIGN KEY REFERENCES dbo.instructions_types(instruction_id)
);
CREATE TABLE staff_movements(
    movement_id             INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    movement_date           DATE NOT NULL DEFAULT GETDATE(),
    movement_employee_id    INT NOT NULL FOREIGN KEY REFERENCES dbo.employees(employee_id)
                            ON DELETE CASCADE
                            ON UPDATE CASCADE,
    movement_type           INT NOT NULL FOREIGN KEY REFERENCES dbo.staff_movement_types(move_type_id)
);
```

Рисунок 4 – SQL скрипт для создания основных таблиц

```
USE HRDepartment;
CREATE TABLE instructions_types(
    instruction_id          INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    instruction_name        VARCHAR(30) NOT NULL,
    instruction_description VARCHAR(120)
);
CREATE TABLE departments(
    department_id           INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    department_name         VARCHAR(30) NOT NULL,
    department_description  VARCHAR(120)
);
CREATE TABLE education_levels(
    edu_level_id            INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    edu_level_name          VARCHAR(30) NOT NULL,
    edu_level_description   VARCHAR(120)
);
CREATE TABLE martial_statuses(
    mart_status_id          INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    mart_status_name        VARCHAR(30) NOT NULL,
    mart_status_description VARCHAR(120)
);
CREATE TABLE staff_movement_types(
    move_type_id            INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    move_type_name          VARCHAR(30) NOT NULL,
    move_type_description   VARCHAR(120)
);
CREATE TABLE staff_positions(
    position_id             INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    position_name           VARCHAR(30) NOT NULL,
    position_description    VARCHAR(120)
);
CREATE TABLE staff_work_relations(
    work_relation_id          INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
    work_relation_name        VARCHAR(30) NOT NULL,
    work_relation_description VARCHAR(120)
);
GO
```

Рисунок 5 – SQL скрипт для создания вспомогательных таблиц

Для облегчения тестирования приложения БД была заполнена тестовыми данными через SQL скрипты, представленные на рисунках 6 - 8.

```
INSERT INTO employees VALUES
('Тарарыев', 'Матвей', 'Сергеевич', '+7(920)576-61-53', 'mr.tarariev@yandex.ru', '2004-09-19',
(SELECT department_id FROM departments WHERE department_name LIKE 'ОРиВНГиПП'),
(SELECT position_id FROM staff_positions WHERE position_name LIKE 'РазраБотчик ПО'),
(SELECT work_relation_id FROM staff_work_relations WHERE work_relation_name LIKE 'Штатный'),
(SELECT edu_level_id FROM education_levels WHERE edu_level_name LIKE 'Бакалавр'),
(SELECT mart_status_id FROM martial_statuses WHERE mart_status_name LIKE 'Холост/Не замужем'),
0, 0),
('Андропов', 'Сергей', 'Андреевич', DEFAULT, DEFAULT, '1984-07-04',
(SELECT department_id FROM departments WHERE department_name LIKE 'ОСУ'),
(SELECT position_id FROM staff_positions WHERE position_name LIKE 'Системный администратор'),
(SELECT work_relation_id FROM staff_work_relations WHERE work_relation_name LIKE 'Штатный'),
(SELECT edu_level_id FROM education_levels WHERE edu_level_name LIKE 'Специалитет/Магистратура'),
(SELECT mart_status_id FROM martial_statuses WHERE mart_status_name LIKE 'Женат/Замужем'),
1, 0),
('Казаринов', 'Андрей', 'Юрьевич', DEFAULT, DEFAULT, '2005-03-21',
(SELECT department_id FROM departments WHERE department_name LIKE 'Склад №1'),
(SELECT position_id FROM staff_positions WHERE position_name LIKE 'Зав. складом'),
(SELECT work_relation_id FROM staff_work_relations WHERE work_relation_name LIKE 'Штатный'),
(SELECT edu_level_id FROM education_levels WHERE edu_level_name LIKE 'СПО'),
(SELECT mart_status_id FROM martial_statuses WHERE mart_status_name LIKE 'Холост/Не замужем'),
1, 0),
('Виноградов', 'Дмитрий', 'Анатольевич', DEFAULT, DEFAULT, '2002-05-08',
(SELECT department_id FROM departments WHERE department_name LIKE 'Склад №1'),
(SELECT position_id FROM staff_positions WHERE position_name LIKE 'Грузчик'),
(SELECT work_relation_id FROM staff_work_relations WHERE work_relation_name LIKE 'Штатный'),
(SELECT edu_level_id FROM education_levels WHERE edu_level_name LIKE 'СПО'),
(SELECT mart_status_id FROM martial_statuses WHERE mart_status_name LIKE 'Холост/Не замужем'),
0, 0),
('Татаринцев', 'Роман', 'Игоревич', DEFAULT, DEFAULT, '2006-11-25',
(SELECT department_id FROM departments WHERE department_name LIKE 'Цех №1'),
(SELECT position_id FROM staff_positions WHERE position_name LIKE 'Оператор станка с ЧПУ'),
(SELECT work_relation_id FROM staff_work_relations WHERE work_relation_name LIKE 'Самозанятый'),
(SELECT edu_level_id FROM education_levels WHERE edu_level_name LIKE 'Основное общее'),
(SELECT mart_status_id FROM martial_statuses WHERE mart_status_name LIKE 'Холост/Не замужем'),
0, 0),
('Овчинников', 'Игорь', 'Николаевич', DEFAULT, DEFAULT, '1998-07-13',
(SELECT department_id FROM departments WHERE department_name LIKE 'Цех №1'),
(SELECT position_id FROM staff_positions WHERE position_name LIKE 'Оператор станка с ЧПУ'),
(SELECT work_relation_id FROM staff_work_relations WHERE work_relation_name LIKE 'ГПХ'),
(SELECT edu_level_id FROM education_levels WHERE edu_level_name LIKE 'СПО'),
(SELECT mart_status_id FROM martial_statuses WHERE mart_status_name LIKE 'Женат/Замужем'),
0, 0),
('Кулаков', 'Игорь', 'Викторович', DEFAULT, DEFAULT, '1993-07-14',
(SELECT department_id FROM departments WHERE department_name LIKE 'Администрация'),
(SELECT position_id FROM staff_positions WHERE position_name LIKE 'Директор по производству'),
(SELECT work_relation_id FROM staff_work_relations WHERE work_relation_name LIKE 'Штатный'),
(SELECT edu_level_id FROM education_levels WHERE edu_level_name LIKE 'Специалитет/Магистратура'),
(SELECT mart_status_id FROM martial_statuses WHERE mart_status_name LIKE 'Женат/Замужем'),
1, 0),
('Тарарыев', 'Сергей', 'Владимирович', DEFAULT, DEFAULT, '1980-02-20',
(SELECT department_id FROM departments WHERE department_name LIKE 'Цех №1'),
(SELECT position_id FROM staff_positions WHERE position_name LIKE 'Начальник цеха'),
(SELECT work_relation_id FROM staff_work_relations WHERE work_relation_name LIKE 'Штатный'),
(SELECT edu_level_id FROM education_levels WHERE edu_level_name LIKE 'Специалитет/Магистратура'),
(SELECT mart_status_id FROM martial_statuses WHERE mart_status_name LIKE 'Женат/Замужем'),
1, 0);
```

Рисунок 6 – SQL скрипт для заполнения таблицы сотрудников

```sql
--Заполнение записей о прохождении инструктажей--
INSERT INTO safety_training_records(record_date,record_employee_id, record_instruction_type_id) VALUES
('2010-06-14',
(SELECT employee_id FROM employees WHERE employee_surname='Кулаков'),
(SELECT instruction_id FROM instructions_types WHERE instruction_name='Вводный')
),
('2010-06-14',
(SELECT employee_id FROM employees WHERE employee_surname='Татаринцев'),
(SELECT instruction_id FROM instructions_types WHERE instruction_name='Вводный')
),
('2010-07-03',
(SELECT employee_id FROM employees WHERE employee_surname='Казаринов'),
(SELECT instruction_id FROM instructions_types WHERE instruction_name='Вводный')
),
('2010-07-03',
(SELECT employee_id FROM employees WHERE employee_surname='Андропов'),
(SELECT instruction_id FROM instructions_types WHERE instruction_name='Вводный')
),
('2010-07-10',
(SELECT employee_id FROM employees WHERE employee_surname='Татаринцев'),
(SELECT instruction_id FROM instructions_types WHERE instruction_name='Первичный')
),
('2011-01-02',
(SELECT employee_id FROM employees WHERE employee_surname='Казаринов'),
(SELECT instruction_id FROM instructions_types WHERE instruction_name='Первичный')
);

--Заполнение записей о передвижениях сотрудников--
INSERT INTO staff_movements(movement_date, movement_employee_id, movement_type_id) VALUES
('2010-06-14',
(SELECT employee_id FROM employees WHERE employee_surname='Кулаков'),
(SELECT move_type_id FROM staff_movement_types WHERE move_type_name='Прием')
),
('2010-06-14',
(SELECT employee_id FROM employees WHERE employee_surname='Татаринцев'),
(SELECT move_type_id FROM staff_movement_types WHERE move_type_name='Прием')
),
('2010-06-23',
(SELECT employee_id FROM employees WHERE employee_surname='Тарарыев' AND employee_name='Сергей'),
(SELECT move_type_id FROM staff_movement_types WHERE move_type_name='Прием')
),
('2010-07-03',
(SELECT employee_id FROM employees WHERE employee_surname='Казаринов'),
(SELECT move_type_id FROM staff_movement_types WHERE move_type_name='Прием')
),
('2010-07-03',
(SELECT employee_id FROM employees WHERE employee_surname='Андропов'),
(SELECT move_type_id FROM staff_movement_types WHERE move_type_name='Прием')
),
('2010-07-10',
(SELECT employee_id FROM employees WHERE employee_surname='Тарарыев' AND employee_name='Сергей'),
(SELECT move_type_id FROM staff_movement_types WHERE move_type_name='Перевод')
);
```

Рисунок 7 – SQL скрипт для заполнения таблицы сотрудников

```sql
USE HRDepartment;
INSERT INTO departments VALUES
('ОСУ', 'Отдел систем управления (обслуживает все предприятие)'),
('Администрация', 'Административный отдел предприятия'),
('Цех №1', 'Основной производственный цех'),
('Склад №1', 'Основной склад материалов'),
('ОРиВНТиТП', 'Отдел разработки и внедрения новых технологий и технологических процессов');

INSERT INTO education_levels VALUES
('Докторантура', 'Сотрудник имеет статус доктора наук, высшая ступень образования'),
('Специалитет/Магистратура', 'Высшее образование. Сотрудник имеет статус кандидата наук'),
('Бакалавр', 'Начальная ступень высшего образования'),
('СПО', 'Среднее профессиональное образование'),
('НПО', 'Начальное профессиональное образование'),
('Среднее общее', 'Образование 11 классов'),
('Основное общее', 'Образование 9 классов');

INSERT INTO instructions_types VALUES
('Вводный', 'Проводится с вновь принятыми сотрудниками'),
('Первичный', 'Проводится перед началом самостоятельной работы'),
('Повторный', 'Проводится не реже одного раза в полгода по программам первичного инструктажа на рабочем месте'),
('Внеплановый', 'Проводится в случае каких-либо изменений и происшествий'),
('Целевой', 'Проводится в случаях, когданужно выполнить разовые работы либо ликвидировать аварию');

INSERT INTO martial_statuses VALUES
('Холост/Не замужем', 'Мужчина или женщина никогда не состоял(-а) в официально зарегистрированном браке'),
('Женат/Замужем', 'Семейные отношения, которые зарегистрированы в органах ЗАГСа'),
('Гражданский брак', 'Мужчина и женщина проживают как супруги, но юридически не оформили свои отношения'),
('Вдова/Вдовец', 'Супруги находились в зарегистрированном браке, но один из них скончался'),
('Разведен(-а)', 'Мужчина или женщина были в браке, но расторгли его');

INSERT INTO staff_movement_types VALUES
('Прием', 'Прием работника на работу'),
('Перевод', 'Перевод на другую должность, место работы'),
('Увольнение', 'Увольнение работника по всем причинам'),
('Отпуск', 'Предоставление отпусков'),
('Обучение', 'Обучение, повышение квалификации'),
('Декрет', 'Уход в декретный отпуск'),
('Командирование', 'Направление куда-либо со служебным заданием');

INSERT INTO staff_positions VALUES
('Системный администратор', 'Обслуживает IT инфраструктуру предприятия'),
('Директор по производству', 'Управляет производственными процессами предприятия'),
('Начальник цеха', 'Управляет производственными процессами цеха'),
('Оператор станка с ЧПУ', 'Работает со станком и производит товары'),
('Зав. складом', 'Заведует складом'),
('Грузчик', 'Помогает на складе'),
('Разработчик ПО', 'Разрабатывает ПО для микроконтроллеров');

INSERT INTO staff_work_relations VALUES
('Штатный', 'Сотрудник оформлен по ТК РФ'),
('ИП', 'Договор с индивидуальным предпринимателем'),
('Самозанятый', 'Контракт с самозанятым'),
('ГПХ', 'Оформление по гражданско-правовому договору');
```

Рисунок 8 – SQL скрипт для заполнения вспомогательных таблиц

Когда на предприятие устраивается увольняется или восстанавливается из архива сотрудник вместе с добавлением или изменением информации о нем в системе, также должна добавляться запись о его приеме или увольнении, для этого были созданы соответственно триггеры hiring_employee, firing_employee,

restore_employee, для создания которых применялись SQL скрипты, представленные на рисунке 9.

```
USE HRDepartment
GO
CREATE TRIGGER hiring_employee
ON employees
AFTER INSERT
AS
IF (ROWCOUNT_BIG() = 0)
RETURN;
INSERT INTO dbo.staff_movements(movement_date, movement_employee_id, movement_type_id)
SELECT GETDATE(), employee_id, (SELECT move_type_id FROM staff_movement_types WHERE move_type_name = 'Прием')
FROM inserted;
GO
CREATE TRIGGER firing_employee
ON employees
AFTER UPDATE
AS
IF (ROWCOUNT_BIG() = 0)
RETURN;
IF EXISTS (SELECT employee_id FROM inserted
WHERE employee_in_archive = 1 AND employee_id IN (SELECT employee_id FROM deleted WHERE employee_in_archive = 0))
BEGIN
INSERT INTO dbo.staff_movements(movement_date, movement_employee_id, movement_type_id)
SELECT GETDATE(), employee_id, (SELECT move_type_id FROM staff_movement_types WHERE move_type_name = 'Увольнение')
FROM inserted
WHERE employee_in_archive = 1 AND employee_id IN (SELECT employee_id FROM deleted WHERE employee_in_archive = 0);
END
IF EXISTS (SELECT employee_id FROM inserted
WHERE employee_in_archive = 0 AND employee_id IN (SELECT employee_id FROM deleted WHERE employee_in_archive = 1))
BEGIN
INSERT INTO dbo.staff_movements(movement_date, movement_employee_id, movement_type_id)
SELECT GETDATE(), employee_id, (SELECT move_type_id FROM staff_movement_types WHERE move_type_name = 'Прием')
FROM inserted
WHERE employee_in_archive = 0 AND employee_id IN (SELECT employee_id FROM deleted WHERE employee_in_archive = 1);
END
GO
CREATE TRIGGER restore_employee
ON employees
AFTER UPDATE
AS
IF (ROWCOUNT_BIG() = 0)
RETURN;
IF EXISTS (SELECT employee_id FROM inserted
WHERE employee_in_archive = 0 AND employee_id IN (SELECT employee_id FROM deleted WHERE employee_in_archive = 1))
BEGIN
INSERT INTO dbo.staff_movements(movement_date, movement_employee_id, movement_type_id)
SELECT GETDATE(), employee_id, (SELECT move_type_id FROM staff_movement_types WHERE move_type_name = 'Прием')
FROM inserted
WHERE employee_in_archive = 0 AND employee_id IN (SELECT employee_id FROM deleted WHERE employee_in_archive = 1)
END;
```

Рисунок 9 – SQL скрипт для создания триггеров таблицы employees

Форма и названия полей таблиц физической модели неудобно для восприятия рядовым пользователем, поэтому целесообразно сделать понятные представления этих таблиц. Для этого также использовались SQL скрипты, отрывок представлений для основных таблиц представлен на рисунках 10, для автоматизации создания БД в случае ее удаления.

```
CREATE OR ALTER VIEW all_employees_view
AS
SELECT  employee_id AS Табельный_номер,
        CONCAT(employee_surname, ' ', employee_name, ' ', employee_patronymic) AS ФИО,
        employee_birth_date AS [Дата рождения],
        department_name AS Отдел,
        position_name AS Должность,
        work_relation_name AS [Форма трудовых отношений],
        edu_level_name AS [Образование],
        mart_status_name AS [Семейное положение],
        IIF(employee_have_a_child=1, 'Имеет', 'Не имеет') AS [Наличие ребенка],
        IIF(employee_in_archive=1, 'В архиве', 'Нет') AS [В архиве]
FROM employees
JOIN departments ON employee_department_id = department_id
JOIN staff_positions ON employee_position_id = position_id
JOIN staff_work_relations ON employee_work_relation_id = work_relation_id
JOIN education_levels ON employee_education_level_id = edu_level_id
JOIN martial_statuses ON employee_martial_status_id = mart_status_id
WITH CHECK OPTION;
GO
CREATE OR ALTER VIEW all_staff_movements_view
AS
SELECT  staff_movements.movement_id AS [Код передвижения],
        staff_movements.movement_date AS [Дата передвижения],
        employees.employee_id AS [Табельный номер],
        CONCAT(employee_surname, ' ', employee_name, ' ', employee_patronymic) AS ФИО,
        staff_movement_types.move_type_name AS [Тип передвижения]
FROM staff_movements
    JOIN staff_movement_types ON staff_movements.movement_type_id = staff_movement_types.move_type_id
    JOIN employees ON movement_employee_id = employee_id;
GO
CREATE OR ALTER VIEW all_safety_trainings_records_view
AS
SELECT  record_id AS [Код записи],
        record_date AS [Дата проведения инструктажа],
        employees.employee_id AS [Табельный номер],
        CONCAT(employee_surname, ' ', employee_name, ' ', employee_patronymic) AS ФИО,
        instruction_name AS [Тип инструктажа]
FROM safety_training_records
    JOIN employees ON record_employee_id = employee_id
    JOIN instructions_types ON record_instruction_type_id = instruction_id;
GO
```

Рисунок 10 – часть SQL скрипта для создания представлений основных таблиц

В обязанности сотрудника отдела кадров также входит учет прохождения различных инструктажей, для того чтобы понять, какие сотрудники не прошли обязательные вводные и первичные инструктажи была создана процедура get_employees_without_safe_train_record, SQL скрипт для создания которой представлен на рисунке 11.

```
--Получение перечня сотрудников, которые ни разу не прошли инструктаж--
CREATE OR ALTER PROCEDURE get_employees_without_safe_train_record
AS
    BEGIN
    SELECT * FROM employees_without_safe_train_record_view
    END;
GO
```

Рисунок 11 – часть SQL скрипта для создания представлений основных таблиц

По итогам выполнения данного раздела была спроектирована и разработана база данных для информационной системы «Отдел кадров организации».

Список литературы:

1. Карпова Т. С. Базы данных: модели, разработка, реализация: учебное пособие: Электронный ресурс / Т. С. Карпова. - Москва: ИНТУИТ, 2016. - 241 с. Режим доступа: http://biblioclub.ru/index.php?page=book_red&id=429003&sr=1

2. Корнева, Г. В. Этапы внедрения информационных систем: анализ, проектирование, разработка и реализация / Г. В. Корнева, Е. А. Вострикова // Детерминанты развития экономики и общества в условиях глобальных изменений

: Сборник статей II международной научно-практической конференции, Москва, 25–26 апреля 2024 года. – Курск: ЗАО «Университетская книга», 2024. – С. 319-324. – EDN CCWENP.

3. Кузин, А. В. Основы проектирования баз данных: Учебное пособие для студентов средних специальных учебных заведений, обучающихся по направлению «Информационные системы» / А. В. Кузин. – Москва: Общество с ограниченной ответственностью «Научно-издательский центр ИНФРА-М», 2025. – 229 с. – ISBN 978-5-16-016312-3. – DOI 10.12737/1096072. – EDN QNGJXZ.00:30

4. Чурбанова О.В. Базы данных и знаний. Проектирование баз данных в Microsoft Access: учебно-методическое пособие: Электронный ресурс / О. В. Чурбанова, А. Л. Чурбанов. - Архангельск: САФУ, 2015. - 152 с.

5. Шурига, Н. С. Проектирование информационного ресурса кадрового агентства / Н. С. Шурига, Н. В. Тимофеева // Тенденции развития науки и образования. – 2024. – № 105-14. – С. 100-103. – DOI 10.18411/trnio-01-2024-712. – EDN WAUZYG.