МИНИСТЕРСТВО ОБРАЗОВАНИЯ, НАУКИ И МОЛОДЕЖИ РЕСПУБЛИКИ КРЫМ

Государственное бюджетное образовательное учреждение высшего образования Республики Крым
«Крымский инженерно-педагогический университет имени Февзи Якубова»
(ГБОУВО РК КИПУ имени Февзи Якубова)

Кафедра прикладной информатики

СОГЛАСОВАНО
Руководитель ОПОП
_____ Л.Н. Акимова
(подпись) *(инициалы, фамилия)*
«12 » 02 2025 г.

УТВЕРЖДАЮ
Заведующий кафедрой
_____ З.С. Сейдаметова
(подпись) *(инициалы, фамилия)*
«12 » 02 2025 г.

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ

ПО ВЫПОЛНЕНИЮ КУРСОВОГО ПРОКТА

по дисциплине
БАЗЫ ДАННЫХ

направление подготовки
09.03.03 Прикладная информатика

профиль подготовки «Прикладная информатика в информационной сфере»

Факультет экономики, менеджмента и информационных технологий

Лист согласования

методических рекомендаций

по выполнению курсового проекта по дисциплине «Базы данных» направления подготовки 09.03.03 «Прикладная информатика в информационной сфере».

Составитель
методических рекомендаций _____ О.Е. Первун, доцент, к.пед.н.
(подпись)

Методические рекомендации рассмотрены и одобрены на заседании кафедры Прикладная информатика
(протокол от «10» 02 2025 г. № 8)

Заведующий кафедрой _____ З.С. Сейдаметова
(подпись) *(инициалы, фамилия)*

Методические рекомендации рассмотрены и одобрены на заседании УМК факультета экономики, менеджмента и информационных технологий
(протокол от «14» 03 2025 г. № 6)

Председатель УМК _____ К.М. Османов
(подпись) *(инициалы, фамилия)*

Методические рекомендации рекомендованы к использованию ученым советом факультета экономики, менеджмента и информационных технологий
(протокол от «17» 03 2025 г. № 9)

Председатель ученого совета факультета _____ _____
(подпись) *(инициалы, фамилия)*

СОДЕРЖАНИЕ

ЦЕЛЬ И ЗАДАЧИ КУРСОВОГО ПРОЕКТА

Целью выполнения курсового проекта по дисциплине «Базы данных» является приобретение навыков разработки законченных программных комплексов для решения практических задач в области обработки данных с использованием современных СУБД, систематизация и закрепление теоретических знаний, полученных за время обучения, а также приобретение и закрепление навыков самостоятельной работы.

Курсовой проект выполняется по актуальным вопросам систем обработки данных по индивидуальному или комплексному заданию и оформляется в виде пояснительной записки. В каждом курсовом проекте должны быть проработаны все разделы и части, которые регламентируются настоящей методикой.

ТЕМАТИКА КУРСОВОГО ПРОЕКТИРОВАНИЯ

Тематика курсовых проектов должна соответствовать основным разделам программы дисциплины «Базы данных». Теоретическая часть курсовой работы должна базироваться на лекционном материале дисциплины. Курсовой проект должен содержать углубленную разработку вопросов проектирования баз данных. Тематика курсовых проектов определяется преподавателем.

ПОРЯДОК ВЫПОЛНЕНИЯ КУРСОВОГО ПРОЕКТА

Основная цель процесса проектирования БД состоит в получении такого проекта, который удовлетворяет следующим требованиям:

1. Корректность схемы БД, т.е. база должна быть гомоморфным образом моделируемой ПО, где каждому объекту ПО соответствуют данные в памяти ЭВМ, а каждому процессу – адекватные процедуры обработки данных.
2. Обеспечение ограничений (на объёмы внешней и оперативной памяти и другие ресурсы вычислительной системы).
3. Эффективность функционирования (соблюдение ограничений на время реакции системы на запрос и обновление данных).
4. Защита данных (от сбоев и несанкционированного доступа).
5. Простота и удобство эксплуатации.
6. Гибкость, т.е. возможность развития и адаптации к изменениям ПО и/или требований пользователей.

Удовлетворение первых 4-х требований обязательно для принятия проекта.

Процесс проектирования БД включает в себя следующие этапы:

1. Информационно-логическое (инфологическое) проектирование.
2. Определение требований к операционной обстановке, в которой будет функционировать информационная система.
3. Выбор СУБД и других инструментальных программных средств.
4. Логическое проектирование БД.
5. Физическое проектирование БД.

Этап 1. Инфологическое проектирование

Инфологический подход не предоставляет формальных способов моделирования реальности, однако он закладывает основы методологии проектирования БД.

Первой задачей инфологического проектирования является определение ПО системы, позволяющее изучить информационные потребности будущих пользователей. Другая задача этого этапа – анализ ПО, который призван сформировать взгляд на ПО с позиций сообщества будущих пользователей БД, т.е. инфологической модели ПО. Анализ ПО выполняется разработчиком логической базы данных – специалистом в данной ПО.

Инфологическая модель ПО представляет собой описание структуры и динамики ПО, характера информационных потребностей пользователей системы в терминах, понятных пользователю и независимых от реализации системы. Более того, инфологическая модель ПО не должна зависеть от модели данных, которая будет использована при создании БД.

Обычно описание ПО выражается в терминах не отдельных объектов и связей между ними, а их типов, связанных с ними ограничений целостности и тех процессов ПО, которые приводят к переходу ПО из одного состояния в другое. Такое описание может быть представлено любым способом, допускающим однозначную интерпретацию.

В простых случаях описание ПО представляется на естественном языке, в более сложных используется также математический аппарат: таблицы, диаграммы, графы и т.п.

При выполнении курсовой работы необходимо придерживаться следующих обозначений:

1. Имя отношения выделяется курсивом и подчеркиванием и пишется прописными буквами, например: *СОТРУДНИКИ*.
2. Имя атрибута отношения выделяется курсивом и подчеркиванием и пишется с большой буквы, например: *Оклад*.
3. Ключевые атрибуты отношения выделяются полужирным шрифтом, например: ***Табельный номер***.
4. Имя связи между отношениями выделяется курсивом и подчеркиванием и пишется строчными буквами, например: *работает*.

Построение инфологической модели базы данных необходимо производить с использованием метода "сущность–связь".

Необходимо разбить ПО на ряд локальных областей, каждая из которых (в идеале) включает в себя информацию, достаточную для обеспечения информационных потребностей одной группы будущих пользователей или решения отдельной задачи. Каждое локальное представление моделируется отдельно, а затем выполняется их объединение. Выбор локального представления зависит от масштабов ПО. Обычно ПО разбивается на локальные области так, чтобы каждая из них соответствовала отдельному внешнему приложению и содержала 6-7 сущностей (т.е. объектов, о которых в системе будет накапливаться информация).

Для каждой сущности определяются атрибуты, которые делятся на два типа: *идентифицирующие* и *описательные*. Идентифицирующие атрибуты входят в состав ключа (или ключей) и позволяют однозначно распознавать экземпляры сущности. Первичный ключ базовой сущности не может содержать неопределённые значения атрибутов (null). Первичный ключ должен включать в свой состав минимально необходимое для идентификации количество атрибутов. Описательные атрибуты заключают в себе свойства сущности, интересующие пользователей.

Спецификация атрибута состоит из его названия, указания типа данных и описания ограничений целостности – множества значений, которые может принимать данный атрибут.

Далее осуществляется спецификация связей: выявляются связи между сущностями внутри локального представления. Каждая связь именуется. Кроме спецификации связей типа "сущность – сущность", выполняется спецификация связей типа "сущность – атрибут" и "атрибут – атрибут" для отношений между атрибутами, которые относятся к одной и той же сущности или к одной и той же связи типа "сущность – сущность".

При объединении проектировщик может формировать конструкции, производные по отношению к тем, которые были использованы в локальных представлениях. Цель введения подобных абстракций:

- объединение в единое целое фрагментарных представлений о различных свойствах одного и того же объекта;
- введение абстрактных понятий, удобных для решения задач системы, установление их связи с более конкретными понятиями модели;
- образование классов и подклассов подобных объектов (например, класс "изделие" и подклассы типов изделий, производимых на предприятии).

При небольшом количестве локальных областей (не более пяти) объединение выполняется за один шаг. В противном случае обычно выполняют бинарное объединение. При объединении представлений используют три основополагающие концепции:

1. **Идентичность**. Два или более элементов модели идентичны, если они имеют одинаковое семантическое значение.

2. **Агрегация**. Позволяет рассматривать связь между элементами как новый элемент. Например, связь *экзамен* между сущностями *СТУДЕНТ*, *ДИСЦИПЛИНА*, *ПРЕПОДАВАТЕЛЬ* может быть представлена агрегированной сущностью *ЭКЗАМЕН* с атрибутами *Название дисциплины*, *Фамилия преподавателя*, *Фамилия студента*, *Оценка*.

3. **Обобщение**. Позволяет образовывать многоуровневую иерархию обобщений.

По завершении объединения результаты проектирования представляют собой концептуальную инфологическую модель ПО. Модели локальных представлений – это внешние инфологические модели.

БД publications должна хранить сведения о печатных изданиях, а также ссылки на интересные ресурсы в Internet. И те, и другие источники информации будут касаться одной темы, а именно "баз данных". Попробуем выделить интересующие нас сущности и определить связи между ними.

Прежде всего займемся понятием "печатное издание". Что это такое? Мы знаем, что объект "печатное издание" воплощается в виде книги, которую можно полностью описать с помощью следующих характеристик: название, автор, год издания и издатель (издательство). Можно ли на основании этого ввести сущность "книга", а названные характеристики определить в качестве ее атрибутов? Прежде чем сделать это рассмотрим более внимательно отношения между книгой и ее харакетристиками:

Один автор может написать несколько книг, и, в то же время, одна книга может быть написана несколькими авторами. Следовательно, "книга" и "автор" в данном случае выступают как различные сущности, объединяемые связью N : M. Для того, чтобы определить класс принадлежности сущностей в связи, отметим, что книг без авторов не бывает, как и авторов без книг. Значит, каждая сущность должна иметь обязательный класс принадлежности (кардинальность связи(1,N) : (1,N)).

Точно так же один издатель может издавать сразу несколько книг, но каждая конкретная книга издается только в одном месте. Следовательно, мы должны ввести сущность "издатель", ассоциируемую с "книгой" связью типа 1 : N. Т.к. каждая книга кем-то издана, класс принадлежности сущности "издатель" в данной связи будет (1,1), но в то же время мы допускаем хранение сведений об издательствах, чьих книг в нашей базе данных пока нет. Соответственно, класс принадлежности сущности "книга" в этой связи (0,N).

По поводу характеристики книги "название" можно сказать следующее: как правило авторы, пишущие на одну тему, стараются придумывать для своих призведений оригинальные названия. Поэтому, можно уверенно предположить, что каждое название обязательно связано только с одной книгой (и каждая книга имеет только одно название). Следовательно, "название" нужно оставить в списке атрибутов "книги".

Те же рассуждения можно повторить и для характеристики "год издания". Ее мы тоже оставим в списке атрибутов "книги".

Таким образом, мы определили, что у сущности "книга" имеется два атрибута "название" и "год издания". Как уже говорилось, название, скорее всего, будет однозначно определять данную книгу, чего не скажешь о годе издания. Поэтому объявим ключом сущности атрибут "название" (или "имя_книги").

Что касается всех возможных авторов, то нас интересует только одна их харакетристика - имя. Поэтому, сущность "автор" имеет только один атрибут "имя_автора", который и является ключом.

С сущностью "издатель" дел обстоит несколько сложнее. Практически все крупные издательства имеют сейчас собственные web-страницы, которые могут содержать информацию полезную для пользователей проектируемой базы данных. Поэтому, нужно рассмотреть две характеристики этого объекта: "имя_издателя" и "URL" (uniform resource locator - универсальный указатель ресурсов, с помощью которого в Internet определяется путь к web - странице). Ясно, что каждый издатель имеет уникальное имя и уникальный url, но прежде чем внести их в список атрибутов, вспомним, что наша база данных должна также содержать ссылки и на другие Internet-ресурсы. Возможно, при дальнейшем анализе возникнет необходимость во введении отдельной сущности "URL". Поэтому "имя_издателя" внесем в список атрибутов сущности "издатель", а "URL" будем считать атрибутом отдельной сущности "web - страница", ассоциируемой с "издателем" связью (1,1):(1,1).

Теперь настала пора занятся объектом "ресурс Internet". Его мы можем описать с помощью понятий "имя ресурса", "url", "автор". Внимательно рассмотрев связи этих

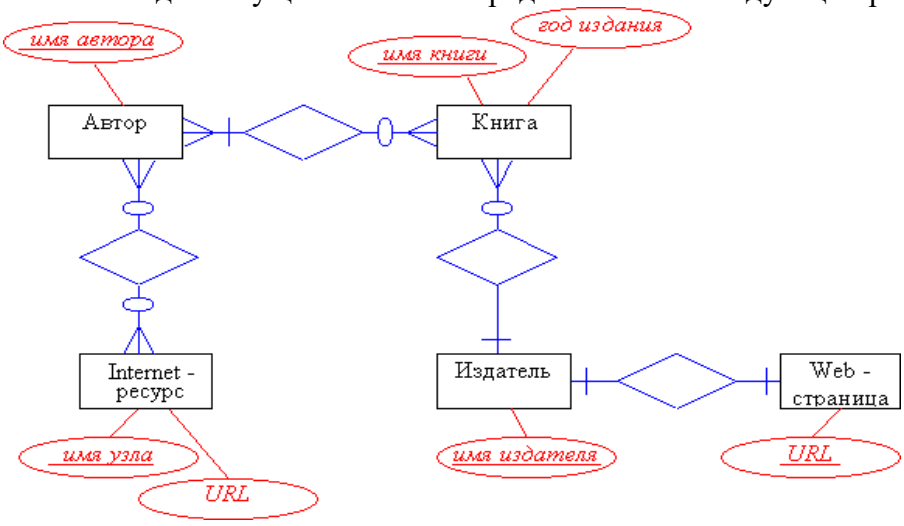
понятий с описываемым объектом, можно прийти к заключению, что "имя_ресурса" и "url" однозначно с ним связаны, т.е. являются атрибутами. В то же время, "автор" является отдельной сущностью (один ресурс может иметь много авторов, и один автор может быть создателем многих web - страниц). Т.к. мы уже ранее вввели сущность "автор" просто определим характеристики ее связи с сущностью "Internet-ресурс". Из сказанного выше следует, что эти сущности объединяются связью n : m, в то же время, автор какой-либо книги может не иметь собственной web - страницы, а авторы некоторых Internet ресурсов не указывают своих имен (т.е. можно формально сказать, что эти ресурсы не имеют авторов). Следовательно, класс принадлежности обеих сущностей будет необязательным.

Прежде чем объявить нашу модель готовой, проверим еще раз определение каждой сущности. Внимательный анализ покажет, что построенная модель имеет несколько ошибок:

Сущность "автор" имеет обязательный класс принадлежности в связи с сущностью "книга". Это означает, что мы не сможем добавить в базу данных сведения о человеке, который создал собственный web - сайт, но не написал ни одной книги. Для того, чтобы устранить это ограничение изменим класс принадлежности сущности "книга" в рассматриваемой связи "автор" - "книга" на необязательный.

При анализе объекта "издатель" мы предположили, что сущность "web-страница" может быть объединена с сущностью "Internet-ресурс". Однако, мы видим, что эти сущности имеют разный набор атрибутов, следовательно, выполнить такое объединение нельзя. Вспомним, что в противном случае, предполагалось единственный атрибут сущности "web - страница" присоединить к атрибутам сущности "издатель". Тем не менее, не будем этого делать, в следующем разделе мы увидим, что с помощью правил порождения реляционных отношений из модели "сущность-связь" в том и в другом случае мы получим одинаковый результат.

Готовая модель "сущность-связь" представлена на следующем рисунке:



На этапе анализа ПО также необходимо решить следующие задачи:

1. Определить правила (ограничений целостности), которым должны удовлетворять сущности ПО, атрибуты сущностей и связи между ними. Часть этих правил реализуется в схеме базы данных (возможности реализации ограничений целостности в схеме БД определяются моделью данных той СУБД, которая будет выбрана для реализации проекта). Остальные правила реализуются с помощью программного обеспечения.

2. Выделить группы пользователей системы. Каждая группа выполняет определённые задачи и обладает разными правами доступа к системе.

3. Создать внешнюю спецификацию тех функций (процессов), которые эта система будет выполнять. Например, для той же библиотечной системы это задачи поиска книг (по определённым критериям), выдачи/приёма книг, определение списка должников и т.д.

Этап 2. Определение требований к операционной обстановке

На этом этапе производится оценка требований к вычислительным ресурсам, необходимым для функционирования системы, выбор типа и конфигурации ЭВМ, типа и версии операционной системы.

Выбор зависит от таких следующих показателей:

- примерный объём данных в БД;
- динамика роста объёма данных;
- характер запросов к данным (извлечение и обновление отдельных записей, групп записей, обработка отдельных отношений или соединение отношений);
- интенсивность запросов к данным по типам запросов;
- требования к времени отклика системы по типам запросов.

Этап 3. Выбор СУБД и инструментальных программных средств

Выбор СУБД является одним из важнейших моментов в разработке проекта БД, так как он принципиальным образом влияет на весь процесс проектирования БД и реализации информационной системы.

Теоретически при осуществлении этого выбора нужно принимать во внимание десятки факторов. Но на практике разработчики руководствуются лишь собственной интуицией и несколькими наиболее важными критериями, к которым, в частности, относятся:

- тип модели данных, которую поддерживает данная СУБД, адекватность модели данных структуре рассматриваемой ПО;
- характеристики производительности СУБД;
- запас функциональных возможностей для дальнейшего развития информационной системы;
- степень оснащенности СУБД инструментарием для персонала администрирования данными;
- удобство и надежность СУБД в эксплуатации;
- стоимость СУБД и дополнительного программного обеспечения.

В связи с тем. что в данной работе СУБД задается в задании, на данном этапе решается задача выбора инструментального ПО.

Этап 4. Логическое проектирование БД

На этапе логического проектирования разрабатывается логическая структура БД, соответствующая инфологической модели ПО. Решение этой задачи существенно зависит от модели данных, поддерживаемой выбранной СУБД. Результатом выполнения этого этапа являются схемы БД концептуального и внешнего уровней архитектуры, составленные на языках определения данных (DDL) выбранной СУБД.

При этом разработка логической структуры базы данных должна производиться с учетом соблюдения условий нормализации отношений.

На начальном шаге логического проектирования таблцы строятся такм образом, чтобы минимизировать количество таблиц в базе данных. Очевдино, что при таком подходе отношения не будут находиться ни в одной из нормальных форм, либо находиться в 1й нормальной форме. Необходимо последовательно привести каждую таблицу к 4й нормальной форме.

Ниже рассматривается отношение ***КНИГИ*** (табл. 3.1) и последовательность его приведения к 4й нормальной форме.

Id – идентификатор (первичный ключ),
Code – шифр рубрики,
Theme– название рубрики,
Title – название книги,
Author– автор,
Editor – редактор,
Type – тип издания (учебник, учебное пособие, сборник и.т.п.),
Year – год издания,
Pg – количество страниц.

Таблица 3.1. Исходное отношение *КНИГИ*

ID	*Code*	*Theme*	*Author*	*Title*	*Editor*	*Type*	*Year*	*Pg*
200	681.3	ПО ВТ	Бочков С. Субботин Д.	Язык СИ	Садчиков П.	учебник	1990	384
100	681.3	ПО ВТ	Джехани Н.	Язык АДА		учебник	1960	552
300	621.5	МО	Крон Г.	Диакоптика	Баранов А.	учебник	1972	544
876	007	ИИ	Гик Е.Я.	Шахматы и математика	Кикоин И. Капица С.	учебное пособие	1983	176
440	32.97	ВТ		ПУ для ПЭВМ	Витенберг А.	справочник	1992	208
385	001.8	Инфор-матика	Фролов Г. Кузнецов Э.	Элементы информатики	Храмов А. Рожков П.	учебное пособие	1989	304

Примечание. В таблице 3.1 используются следующие сокращения:
ВТ – вычислительная техника;
ПО ВТ – программное обеспечение вычислительной техники;
МО – математическое обеспечение;
ИИ – искусственный интеллект.

Первая нормальная форма (1НФ).

Отношение приведено к 1НФ, если все его атрибуты простые.

Отношение *КНИГИ* содержит сложные атрибуты *Author* ("Авторы") и *Editor* ("Редакторы"). Для приведения к 1НФ требуется сделать ключ отношения составным – атрибуты *ID*, *Author* и *Editor* (табл. 3.2).

Таблица 3.2. Отношение *КНИГИ*, приведённое к 1НФ

ID	*Code*	*Theme*	***Author***	*Title*	***Editor***	*Type*	*Year*	*Pg*
200	681.3	ПО ВТ	Бочков С.	Язык СИ	Садчиков П.	учебник	1990	384
200	681.3	ПО ВТ	Субботин Д.	Язык СИ	Садчиков П.	учебник	1990	384
100	681.3	ПО ВТ	Джехани Н.	Язык АДА		учебник	1960	552
300	621.5	МО	Крон Г.	Диакоптика	Баранов А.	учебник	1972	544
876	007	ИИ	Гик Е.Я.	Шахматы и математика	Кикоин И.	учебное пособие	1983	176
876	007	ИИ	Гик Е.Я.	Шахматы и математика	Капица С.	учебное пособие	1983	176
440	32.97	ВТ		ПУ для ПЭВМ	Витенберг А.	Спра-вочник	1992	208
385	001.8	Инфор-матика	Фролов Г.	Элементы информатики	Храмов А.	учебное пособие	1989	304
385	001.8	Инфор-матика	Кузнецов Э.	Элементы информатики	Рожков П.	учебное пособие	1989	304

Вторая нормальная форма (2НФ).

Введём понятие **функциональной зависимости**. Пусть X и Y – атрибуты (группы атрибутов) некоторого отношения. Говорят, что Y функционально зависит от X, если в любой момент времени каждому значению X=x соответствует единственное значение Y=y (X□Y). (При этом любому значению Y=y может соответствовать несколько значений X=(x_1, x_2,…)).

Атрибут X в функциональной зависимости X□Y называется *детерминантом* отношения.

В нормализованном отношении все неключевые атрибуты функционально зависят от ключа отношения. Говорят, что неключевой атрибут функционально полно зависит от составного ключа, если он функционально зависит от ключа, но не находится в функциональной зависимости ни от какой части составного ключа.

Отношение находится во 2НФ, если оно приведено к 1НФ и каждый неключевой атрибут функционально полно зависит от составного ключа.

Для того чтобы привести отношение ко 2НФ, нужно:

- построить его проекцию, исключив атрибуты, которые не находятся в функционально полной зависимости от составного ключа;
- построить дополнительные проекции на часть составного ключа и атрибуты, функционально зависящие от этой части ключа.

Ключом отношения *КНИГИ* (табл. 3.2) является комбинация полей (***ID, Author, Editor***). Все поля, не входящие в состав ключа, зависят только от идентификатора книги. Поэтому отношение должно быть разбито на два: *КНИГИ* (табл. 3.3) и *КНИГИ–АВТОРЫ–РЕДАКТОРЫ* (табл. 3.4). Эти отношения связаны по внешнему ключу, которым является поле ID.

Таблица 3.3. Отношение *КНИГИ*, приведённое к 2НФ

ID	*Code*	*Theme*	*Title*	*Type*	*Year*	*Pg*
200	681.3	ПО ВТ	Язык СИ для ПК	Учебник	1990	384
100	681.3	ПО ВТ	Язык АДА	Учебник	1960	552
300	621.5	МО	Диакоптика	Учебник	1972	544
876	007	ИИ	Шахматы и математика	учебное пособие	1983	176
440	32.97	ВТ	ПУ для ПЭВМ	Справочник	1992	208
385	001.8	Информатика	Элементы информатики	учебное пособие	1989	304

Таблица 3.4. Отношение *КНИГИ–АВТОРЫ–РЕДАКТОРЫ* (2НФ)

ID	***Author***	***Editor***
200	Бочков С.	Садчиков П.
200	Субботин Д.	Садчиков П.
100	Джехани Н.	
300	Крон Г.	Баранов А.
876	Гик Е.Я.	Кикоин И.
876	Гик Е.Я.	Капица С.
440		Витенберг А.
385	Фролов Г.	Храмов А.
385	Кузнецов Э.	Рожков П.

Третья нормальная форма (3НФ).

Рассмотрим понятие **транзитивной зависимости**. Пусть X, Y, Z – атрибуты некоторого отношения. При этом X□ Y и Y□ Z, но обратное соответствие отсутствует, т.е. Z не зависит от Y или Y не зависит от X. Тогда говорят, что Z транзитивно зависит от X (X□□ Z).

Отношение находится в 3НФ, если оно находится во 2НФ и в нем отсутствуют транзитивные зависимости.

Для отношения *КНИГИ* (табл. 3.3) атрибут *Theme* зависит от атрибута *Code*, а не от ключа (хотя название рубрики, естественно, соответствует её шифру). Поэтому для приведения отношения к 3НФ (табл. 3.5) нужно выделить из него ещё одно отношение *РУБРИКАТОР* (табл. 3.6).

Таблица 3.5. Отношение *КНИГИ*, приведённое к 3НФ

ID	*Code*	*Title*	*Type*	*Year*	*Pg*
200	681.3	Язык СИ для ПК	Учебник	1990	384
100	681.3	Язык АДА	Учебник	1960	552
300	621.5	Диакоптика	Учебник	1972	544
440	32.97	ПУ для ПЭВМ	Справочник	1992	208
876	007	Шахматы и математика	учебное пособие	1983	176
385	001.8	Элементы информатики	учебное пособие	1989	304

.

Таблица 3.6. Отношение *РУБРИКАТОР*, приведённое к 3НФ

Code	*Theme*

681.3	ПО ВТ
621.5	МО
007	ИИ
32.97	ВТ
001.8	Информатика

Четвертая нормальная форма (4НФ).

Введём понятие **многозначной зависимости**. Многозначная зависимость существует, если заданным значениям атрибута X соответствует множество, состоящее из нуля (или более) значений атрибута Y (X–»Y). Если в отношении присутствуют многозначные зависимости, то схема отношения должна находиться в 4НФ.

Различают тривиальные и нетривиальные многозначные зависимости. **Тривиальной** называется такая многозначная зависимость X–»Y, для которой Y Ì X или X U Y = R, где R – рассматриваемое отношение. Тривиальная многозначная зависимость не нарушает 4НФ. Если хотя бы одно из двух этих условий не выполняется (т.е. Y не является подмножеством X или X U Y состоит не из всех атрибутов R), то такая многозначная зависимость называется нетривиальной.

Отношение находится в 4НФ, если оно находится в 3НФ и в нем отсутствуют нетривиальные многозначные зависимости.

Для отношения *КНИГИ–АВТОРЫ–РЕДАКТОРЫ* (табл. 3.4) атрибуты *Author* и *Editor* зависит образуют две многозначные зависимости от первичного ключа, и при этом значения этих атрибутов не зависят друг от друга. Поэтому для приведения отношения к 4НФ нужно разбить его на два отношения *КНИГИ–АВТОРЫ* и *КНИГИ–РЕДАКТОРЫ* (табл. 3.7, 3.8).

Таблица 3.7. Отношение *КНИГИ–АВТОРЫ* (4НФ)

ID	***Author***
200	Бочков С.
200	Субботин Д.
100	Джехани Н.
300	Крон Г.
876	Гик Е.Я.
385	Фролов Г.
385	Кузнецов Э.

.

Таблица 3.8. Отношение *КНИГИ–РЕДАКТОРЫ* (4НФ)

ID	***Editor***
200	Садчиков П.
300	Баранов А.
876	Кикоин И.
876	Капица С.
440	Витенберг А.
385	Храмов А.
385	Рожков П.

Этап 5. Физическое проектирование БД

Этап физического проектирования заключается в увязке логической структуры БД и физической среды хранения с целью наиболее эффективного размещения данных, т.е. отображении логической структуры БД в структуру хранения. Решается вопрос размещения хранимых данных в пространстве памяти, выбора эффективных методов доступа к различным компонентам "физической" БД. Результаты этого этапа документируются в форме схемы хранения на языке определения хранимых данных. Принятые на этом этапе решения оказывают определяющее влияние на производительность системы.

Фактически проектирование БД имеет итерационный характер. В процессе функционирования системы становится возможным измерение её реальных характеристик, выявление "узких" мест. И если система не отвечает предъявляемым к ней требованиям, то обычно она подвергается реорганизации, т.е. модификации первоначально созданного проекта.

Важным шагом на данном этапе необходимо проработать вопрос использования индексов.

В системах, поддерживающих язык SQL, индекс создаётся командой CREATE INDEX. Индексы повышают производительность запросов, которые выбирают относительно небольшое число строк из таблицы. Для определения целесообразности создания индекса нужно проанализировать запросы, обращённые к таблице, и распределение данных в индексируемых столбцах.

Но даже при наличии такой возможности система не всегда обращается к индексу. Очевидно, что если запрос выбирает больше половины записей отношения, то извлечение данных через индекс потребует больше времени, чем последовательная обработка данных. В подобных случаях использование индекса нецелесообразно.

Обращение к составному индексу возможно только в том случае, если в условиях выбора участвуют столбцы, представляющие собой лидирующую часть составного индекса. Например, если индекс строится по столбцам (X, Y, Z), то обращение к индексу будет происходить в тех случаях, когда в условии запроса участвуют столбцы XYZ, XY или X.

При создании индекса большое значение имеет понятие селективности. **Селективность** определяется процентом строк, имеющих одинаковое значение индексируемого столбца: чем выше этот процент, тем меньше селективность.

Выбор индексируемых столбцов определяется следующими соображениями:

- В первую очередь выбираются столбцы, которые часто встречаются в критериях поиска.
- Стоит индексировать столбцы, которые используются для соединения таблиц или являются внешними ключами. В последнем случае наличие индекса позволяет обновлять строки подчиненной таблицы без блокировки основной таблицы, когда происходит интенсивное конкурентное обновление связанных между собою таблиц.
- Нецелесообразно индексировать столбцы с низкой селективностью. Если селективность столбца низкая, то индексирование проводится только в том случае, если выборка чаще производится по редко встречающимся значениям.
- Не индексируются столбцы, которые часто обновляются, т.к. команды обновления ведут к потере времени на обновление индекса.
- Не индексируются столбцы, которые часто используются как аргументы функций или выражений: как правило, такие функции не позволяют использовать индекс.

Необходимо произвести экспериментальные исследования, показывающие повышение (или снижение) быстродействия индексированной базы данных при выборке/добавлении/удалении данных. Исследования производить многократным выполнением повторяющихся запросов со случайными параметрами. На основании результатов исследования сделать выводы о необходимости индексирования тех или иных столбцов таблиц базы данных.

СОДЕРЖАНИЕ И ОБЪЕМ КУРСОВОГО ПРОЕКТА

Курсовой проект содержит пояснительную записку и электронную реализацию в виде программы и данных.

Пояснительная записка в объеме 20-30 страниц должна содержать основной текст (собственно работу), графические материалы (иллюстрации) и, при необходимости, приложения - разработанную программу с исходным текстом на бумажном и/или электронном носителе, исходные данные и результаты расчетов, алгоритмы, модели, структуры.

Пояснительная записка включает основные разделы в указанной последовательности:

•титульный лист;

•бланк задания, подписанный руководителем и заведующим кафедрой;

•оглавление, включающее наименование всех разделов и пунктов с указанием номеров страниц;

•введение, в котором обосновывается актуальность темы, указываются цель и задачи исследований;

•глава I и подглавы (анализ литературы);

•глава II и подглавы (разработка БД);

•заключение с краткими выводами по результатам работы и предложениями по их использованию;

•список литературы;

•приложения (листинг программы с подробными комментариями и т.п.).

СОДЕРЖАНИЕ РАЗДЕЛОВ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ

Титульный лист должен соответствовать установленному образцу.

Задание на курсовую работу является исходным документом для разработки проекта и должно быть выполнено по установленной форме. Задание заполняется руководителем курсовой работы, согласуется со студентом, выполняющим проект, и утверждается заведующим кафедрой. В задании приводится график выполнения работы.

Оглавление (содержание) включает наименование всех разделов курсовой работы, а также подразделов и пунктов, если они имеют наименование. Для каждого раздела, подраздела и пункта указывается номер страницы.

Введение содержит постановку задачи, анализ цели разработки базы данных, обосновывается актуальность темы. Во введении дается краткий анализ возможных методов решения поставленной задачи, а также анализируются ограничения и требования к программе. Объем введения - 2-3 страницы.

Основная часть состоит из глав, в которых рассматривается существо проблемы. Приводится описание предметной области. Анализируются информационные взаимосвязи предметной области. Дается аналитический обзор возможностей построения программы базы данных. Приводится обоснование модели базы данных и СУБД для разработки программы.

В основной части приводится описание структур данных, используемых в программе, блок-схема программы и подробные алгоритмы работы каждой подпрограммы. Листинг программы с подробными комментариями выносится в приложение.

Заключение должно содержать качественную оценку разработанной программы, а также ее соответствия заданию на курсовой проект. Здесь нужно указать размер программы, временные характеристики ее работы. В этом разделе приводятся ограничения на использование программы: требования к операционной системе и аппаратуре ЭВМ.

Список использованной литературы содержит перечень источников, использованных при выполнении курсовой работы. Указываются только те источники, на которые имеются ссылки в тексте пояснительной записки.

Приложение содержит вспомогательный материал (листинги программ, инструкции по ипользованию программами и т.п.).

СОДЕРЖАНИЕ РАЗДЕЛОВ ОСНОВНОЙ ЧАСТИ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ

Основная часть пояснительной записки к курсовому проекту должна содержать следующие разделы:

1. Теоретическая часть

1.1. Описание задачи

1.2. Инфологическое проектирование

1.3. Выбор СУБД

1.4. Логическое проектирование

2. Проектирование задачи (физическое проектирование)

2.1. Проектирование базы данных.

2.1.1. Создание таблиц и заполнения их тестовой информацией.

2.1.2. Определение условий целостности данных и разработка мероприятий по контролю достоверности данных и обеспечению защиты от несанкционированного доступа.

2.2.Проектирование форм.

2.3.Проектирование отчетов.

2.3.1. Обоснование и определение необходимого набора отчетов, их содержание.

2.4.Проектирование меню проекта.

3. Программирование программной оболочки управления базами данных.

4. Разработка системы оперативной справки.

5. Создание инсталляционного пакета проекта.

6. Заключение

7. Список использованных источников

ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ КУРСОВОГО ПРОЕКТА

Пояснительная записка представляется отпечатанной на одной стороне листа формата А4 (210х297 мм). Формат: полуторный межстрочный интервал, шрифт Times New Roman Cyr 14 пт, с выравниванием по ширине. Поля (мм): верхнее - 20, нижнее - 20, левое - 20, правое -15. Электронный вариант пояснительной записки должен быть представлен файлом формата MS Word. Название файла должно состоять из кода группы и фамилии студента набранных латинскими буквами без пробелов, например *МП 73_IVANOV.DOC.*

Изложение должно быть кратким, четким и вестись от первого лица множественного числа.

Весь текст пояснительной записки делят на разделы. Каждый раздел следует начинать с новой страницы. Разделы в пределах всей пояснительной записки, а также подразделы и пункты имеют порядковые номера, обозначенные арабскими цифрами с точкой в конце, например 1. - первый раздел; 2. - второй раз-дел; 2.1. - первый подраздел второго раздела; 2.1.1. - первый пункт первого подраздела второго раздела. Введение и заключение не нумеруются.

Заголовки разделов пишутся прописными буквами с выравниванием по середине. Заготовки подразделов пишут с абзаца, отступая слева 15 мм, строчными буквами (кроме первой прописной). В заголовке не допускаются переносы слов. Пробелы над заголовками и под ними 1см. Точку в конце заголовка не ставят. Если заголовок состоит из двух предложений, то их разделяют точкой. Заголовок подчеркивать нельзя.

В записке необходимо выдерживать единые обозначения и размерности для используемых параметров, переменных и характеристик.

Иллюстрации (рисунки, схемы, таблицы) располагаются на отдельных страницах пояснительной записки. Иллюстрации в пояснительной записке, кроме таблиц, имеют подпись “рис.”. Номер рисунка состоит из номера раздела и порядкового номера иллюстрации, разделенных точкой. Например, рис. 3.2 (второй рисунок третьего раздела). Иллюстрации снабжаются кратким подрисуночным текстом.

В пояснительной записке рисунки должны быть выполнены на отдельных страницах. Рисунок располагают после той страницы, где на него дана первая ссылка.

Таблицы служат для оформления цифрового материала и приводятся после первого упоминания о них в тексте. На все таблицы должны быть ссылки в тексте, при этом слово “Таблица” в таблице пишут полностью.

Каждая таблица должна иметь заголовок. Заголовок и слово “Таблица” начинают с прописной буквы. Заголовки граф таблиц должны начинаться с прописных букв, подзаголовки со строчных, если они составляют одно предложение с заголовком, и с прописных, если они самостоятельны.

При ссылке в тексте на используемую литературу, указывают порядковый номер по списку литературы, выделенный двум квадратными скобками, например [20], или так [7, 15]. Литературу следует располагать в списке в порядке появления ссылок в тексте. Источник описывается по следующей форме: фамилия и инициалы автора, полное название книги или статьи, место и год издания, объем (для журнала - название журнала, год издания, номер, страницы). Например: **13. Дейт К. Введение в системы баз данных. - М.: Наука, 1980 . - 463 с.**

В пояснительной записке все страницы, в том числе титульный лист, содержание, листы с таблицами, рисунками, графиками, нумеруются. На титульном листе номер не ставят, на последующих страницах номер проставляют в правом верхнем углу арабскими цифрами.

Работа выполняется на одном языке (русском или украинском - по желанию студента).

ЗАЩИТА КУРСОВЫХ ПРОЕКТОВ

Законченный и полностью оформленный курсовой проект, подписанный студентом и консультантами (если они предусмотрены заданием), представляется студентом руководителю за неделю до защиты.

Руководитель обязан проверить пояснительную записку и программную часть. После просмотра и одобрения курсового проекта руководитель подписывает её.

В назначенный для защиты день студент должен явиться заблаговременно и подготовить все материалы (программы и т.п.) для проведения защиты. Если при разработке курсовой работы была использована СУБД отсутствующая в составе программного обеспечения кафедры, студент должен заранее обеспечить её установку.

Продолжительность защиты одного проекта составляет в среднем 15 минут. Для доклада по содержанию курсовой работы студенту предоставляется не более 5-7 минут. В своем докладе студент должен показать роль и назначение разработанной системы, обосновать принятые решения, раскрыть отличия рассматриваемой системы от имевшихся ранее, указать, какие исследования были проведены, и сделать обоснованные выводы по проделанной работе. Во время доклада следует выделять главное, не акцентируя внимание на частных вопросах.

При определении оценки проекта принимается во внимание как полнота охвата темы, уровень теоретической и практической подготовки студента, так и умение эффектно представить результаты своей работы.

Оценка курсового проекта

Каждый курсовой проект с учетом ее содержания оценивается по стобалльной системе. Курсовой проект должен быть написана в сроки, устанавливаемые кафедрой. Проект, который преподаватель признал неудовлетворительным, возвращается для переработки с учетом высказанных в отзыве замечаний. Несвоевременное предоставление курсового проекта на кафедру приравнивается к неявке на экзамен, поэтому студентам, не сдавшим без уважительной причины в срок курсовой проект, ставится неудовлетворительная оценка. Студент, не сдавший курсовой проект в срок, считается имеющим академическую задолженность и не допускается к сдаче экзамена по данной дисциплине.

Срок для проверки курсового проекта – 10 (десять) календарных дней с момента поступления работы на кафедру. Результат проверки курсового проекта фиксируется на титульном листе в оценочной таблице, а текст замечаний, рекомендаций и предложений излагается преподавателем на обратной стороне титульного листа работы.

Критерии оценивания курсового проекта

Работа оценивается по 100-бальной шкале.

ШКАЛА ПЕРЕВОДА БАЛЛОВ В ОЦЕНКИ

Баллы	Баллы по 5-балльной шкале	Оценка
90 – 100	5	Отлично
80 – 89	4+	Очень хорошо
70-79	4	Хорошо
60-69	3+	Удовлетворительно
51-59	3	Достаточно
35-50	2	Неудовлетворительно
1-34	2	Неудовлетворительно

Максимальное количество баллов за курсовую работу равно 100 баллам. Далее результирующий балл переводится в традиционную 5-балльную систему оценок, согласно приведенной выше шкале.

КРИТЕРИИ ОЦЕНИВАНИЯ ЗАЩИТЫ КУРСОВОГО ПРОЕКТА

Раздел	Распределение баллов
1. Постановка задачи	10
2. Проектирование задачи	10
3. Программирование программной оболочки управления базами данных	10
4. Разработка системы оперативной справки	10
5. Создание инсталляционного пакета проекта	10
6. Практическая работа пояснительной записки в MS Word	10
7. Защита курсового проекта	40
ВСЕГО	**100**

Оформление списка использованной литературы

1. Нумерация всей использованной литературы сплошная - от первого до последнего источника.

2. Оформление списка использованной литературы рекомендуется выполнять по принципу алфавитного именного указателя (в общем, алфавите авторов и заглавий) в следующей последовательности:

- литература на русском языке,
- литература на языках народов, пользующихся кириллицей.
- литература на языках народов, пользующихся латиницей

3. Описание источников, включенных в список, выполняется в соответствии с существующими библиографическими правилами.

Литературные и иные источники, на которые имеются отсылки в тексте, указываются в списке использованных источников. Отсылки оформляются единообразно по всему документу – в квадратных скобках указывается порядковый номер затекстовой ссылки в списке использованных источников, например [15]. Если необходимо указать номер страницы, он ставится через запятую после порядкового номера издания, например, [10, с. 37].

В список использованных источников не включаются источники, на которые нет ссылок в тексте и которые фактически не были использованы при написании реферата.

<u>НЕ ПРОВЕРЯЮТСЯ И НЕ ОЦЕНИВАЮТСЯ КУРСОВЫЕ ПРОЕКТЫ :</u>

- ✓ **НЕ СООТВЕТСТВУЮЩИЕ ВЫШЕ ПРИВЕДЕННОЙ СТРУКТУРЕ;**
- ✓ **НЕ ОТФОРМАТИРОВАННЫЕ В СООТВЕТСТВИИ С ВЫШЕ ПРИВЕДЕННЫМИ ТРЕБОВАНИЯМИ;**
- ✓ **НЕ СОДЕРЖАЩИЕ ОГЛАВЛЕНИЯ ИЛИ СПИСКА ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ;**
- ✓ ***СОДЕРЖАЩИЕ*:**
- ➢ **НЕВЕРНО ОФОРМЛЕННЫЙ ТИТУЛЬНЫЙ ЛИСТ;**
- ➢ **НЕВЕРНО ОФОРМЛЕННЫЕ БИБЛИОГРАФИЧЕСКИЕ ССЫЛКИ ИЛИ СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ;**
- ➢ **НЕВЕРНО ОФОРМЛЕННЫЕ ССЫЛКИ НА ИНТЕРНЕТ-РЕСУРСЫ.**

ВСЁ, ЧТО НЕ СООТВЕТСТВУЕТ СТРУКТУРЕ КУРСОВОЙ РАБОТЫ,
СЧИТАЕТСЯ НЕ НАПИСАННЫМ!!!

Литература:

Рекомендуемая литература

а) основная литература:

1. Петров, В. Н. Информационные системы: учебник для вузов / В. Н. Петров. - СПб.: Питер, 2002. - 687 с.

б) дополнительная литература

1. Агальцов, В.П. Базы данных : учебное пособие / В. П. Агальцов. - М. : Мир, 2001. - 375 с.
2. Гарсиа-Молина Г. Системы баз данных: Полный курс / Г. Гарсиа-Молина, Г. Ульман, Д. Уидом ; пер. с англ. и ред. А. С. Варакина. - М. и др. : Вильямс, 2003. - 1083 с.
3. Дейт, К. Введение в системы баз данных / К. Дейт. - М. : Наука, 1980. - 464 с.
4. Диго, С. М. Проектирование и использование баз данных / С. М. Диго. - Москва : Финансы и статистика, - 1995. - 208 с.
5. Дюбуа, П. MySQL:Полное и исчерпывающее руководство по применению и администрированию баз данных MySQL 4, а также программированию приложений / П. Дюбуа ; пер. с англ. и ред. Н. В. Воронина. - Изд. 2-е. - М. : Вильямс, 2004. - 1051с.
6. Карпова, Т. С. Базы данных: Модели, разработка, реализация : учебное пособие / Т. С. Карпова. - СПб. и др. : Питер, 2002 - 303 с.
7. Коннолли, Т. Базы данных : Проектирование, реализация и сопровождение: Теория и практика / Т. Коннолли, К. Бегг, А. Страчан ; под ред. Т. Коннолли, К. Бегг. - Изд. 2-е, испр. и доп. - М. : Вильямс, 2001. - 1111 с.
8. Кренке, Д. Теория и практика построения баз данных / Д. Кренке ; пер. с англ. А. Вахитова. - Изд. 9-е. - СПб. : Питер, 2003. - 799 с.
9. Проектирование структур баз данных : 2 ч. / Т. Тиори, Д. Фрай.- М. : Мир, 1985 - Ч. 1-2.
10. Райордан, Р. Основы реляционных баз данных Р. Райордан; пер. с англ. - М. : Русская Редакция, 2001. - 352 с.
11. Роб, П. Системы баз данных: проектирование, реализация и управление / П. Роб, К. Коронел ; пер. с англ. А. Никифорова. - Изд. 5-е, испр. и доп. - СПб. : БХВ-Петербург, 2004. - 1024 с.
12. Ульман, Д. Основы систем баз данных / Д. Ульман - М. : Финансы и статистика, 1983. - 334 с.
13. Фуфаев, Э.В. Базы данных / Э.В. Фуфаев, Д.Э. Фуфаев. – М.: Академия, 2005. – 320 с.
14. Харрингон, Д. Л. Проектирование реляционных баз данных: Просто и доступно : учебное пособие / Д. Л. Харрингтон ; пер. с англ. И. Дранишников ; под ред. А. Головко. - М. : Лори, 2000. - 230 с.
15. Хомоненко, А. Д. Базы данных : учебник для вузов / А. Д. Хомоненко, В. М. Цыганков, М. Г. Мальцев; под ред. А. Д. Хомоненко. - Изд. 3-е, испр. и доп. - СПб. : КОРОНА принт, 2003. - 665 с.
16. Цикритзис, Д. Модели данных / Д. Цикритзис, Ф. Лоховски - М. : Финансы и статистика, 1985. - 344с.
17. Чекалов, А. П. Базы данных: От проектирования до разработки приложений / А. П. Чекалов. - СПб. : БХВ-Петербург, 2003. – 380 с.
18. Чен, П. Модель “сущность - связь” - шаг к единому представлению о данных. // СУБД. - 1995. № 3. - с. 137 - 158.