МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
“ХАРЬКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ”

Методические указания

к лабораторным работам
«Методы проектирования схем баз данных»
по дисциплине
«Распределённые информационно-аналитические системы»
для студентов специальностей
122 – Компьютерные науки и информационные технологии,
124 – Системный анализ

Утверждено
редакционно-издательским
советом университета,
протокол № 2 от 23.06.2016 г

Харьков
НТУ “ХПИ”
2017

Методические указания к лабораторным работам «Методы проектирования схем баз данных» по дисциплине «Распределенные информационно-аналитические системы» для студентов специальностей 122 – Компьютерные науки и информационные технологии,124 – Системный анализ/ Сост. Ю.Н. Кожин, О.Н. Малых, В.Ф. Прокопенков.– Х.: НТУ “ХПИ”, 2017.–32 с. На рус. яз.

Составители: Ю.Н. Кожин,
О.Н. Малих,
В.Ф. Прокопенков

Рецензент О.В. Горелый

Кафедра системного анализа и информационно-аналитических технологий

ВВЕДЕНИЕ

Целью проектирования схемы базы данных (БД) является адекватное отображение в базе данных сути предметной области, рассматриваемой с точки зрения решения задачи автоматизации.

Основными задачами проектирования схемы БД являются:

- обеспечение хранения в БД всей необходимой информации;
- обеспечение возможности получения данных по всем необходимым запросам;
- сокращение избыточности и дублирования данных;

При проектировании схемы базы данных используется методы ER-модели и методы на основе функциональной и многозначных зависимостей.

Основные преимущества ER-моделей:

- наглядность;
- модели позволяют проектировать базы данных с большим количеством объектов и атрибутов;
- ER-модели реализованы в во многих системах автоматизированного проектирования схем баз данных.

К недостаткам ER-модели можно отнести субъективизм выбора набора сущностей и связей.

Методам функциональных и многозначных зависимостей присущ большой формализм при анализе предметной области, что позволяет разрабатывать схемы базы данных в автоматическом режиме. Недостатком методов является то, что алгоритмы проектирования имеют неполиномиальную сложность.

1. ЛАБОРАТОРНАЯ РАБОТА 1
Построение логической модели базы данных с использованием метода сущность-связь

***Цель работы*:** Изучение методов построения логической модели базы данных выбранной предметной области и преобразование логической модели в физическую.

1.1. Методы проектирования схемы базы данных

В настоящее время существуют две группы методов проектирования схемы базы данных:

- методы с использованием диаграмм сущность-связь;
- методы функциональных и многозначных зависимостей.

Методы, основанные на использовании диаграмм сущность-связь, обеспечивают относительно простой способ формирования схемы базы данных. Однако после проектирования схемы базы данных требуется дополнительная проверка, находятся ли построенные таблицы в соответствующей нормальной форме (обычно схема базы данных приводится к третьей нормальной форме).

Методы с использованием функциональных и многозначных зависимостей являются более формализованными. Однако эти методы могут приводить к построению схемы базы данных с таблицами в труднопонимаемой со стороны пользователя форме. Кроме того, сами алгоритмы проектирования схемы имеют неполиномиальную временную сложность, что ограничивает их использование для баз данных с большим количеством атрибутов.

1.2. Метод сущность-связь

Модели сущность-связь основаны на выделении в предметной области, для которой осуществляется проектирование базы данных, различных типов объектов, информацию о которых требуется хранить в базе данных. Набор однотипных объектов предметной области образует сущность. Меж-

ду сущностями могут быть установлены информационные связи (зависимости), которые также могут быть учтены при проектировании схемы базы данных. Совокупность сущностей и связи между ними составляют информационную модель данных предметной области (*Entity- Relationship* диаграмму).

В настоящее время существует несколько приемов выделения сущностей и связей – нотации Чена, Мартина, Баркера, *IDEF1X* и т.д.

1.2.1. Основные определения

Сущность (Entity) – реальный либо воображаемый объект, имеющий существенное значение для рассматриваемой предметной области. Каждая сущность должна иметь наименование, выраженное существительным в единственном числе. Каждая сущность должна обладать уникальным идентификатором. Каждый экземпляр сущности должен однозначно идентифицироваться и отличаться от всех других экземпляров данного типа сущности.

Атрибут (Attribute) – любая характеристика сущности, значимая для рассматриваемой предметной области и предназначенная для квалификации, идентификации, классификации, количественной характеристики или выражения состояния сущности. Наименование атрибута должно быть выражено существительным в единственном числе.

Связь (Relationship) – поименованная ассоциация между сущностями, значимая для рассматриваемой предметной области.

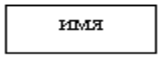
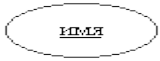
1.2.2. Нотация Чена

В нотации Чена различают зависимые и независимые сущности. Сущность называется независимой, если каждый экземпляр сущности может быть однозначно идентифицирован без определения его отношений с другими сущностями. Сущность называется зависимой от идентификаторов или просто зависимой, если однозначная идентификация экземпляра сущности зависит от его отношения к другой сущности.

Связь соединяется с ассоциируемыми сущностями линиями. Возле каждой сущности на линии, соединяющей ее со связью, цифрами указывается класс принадлежности.

Сущности и связи могут иметь атрибуты. Для каждой сущности находится атрибут (или набор атрибутов), значение которого однозначно определяет экземпляр сущности. Этот атрибут является ключом сущности. Связь также может иметь ключевой атрибут. В ряде случаев для удобства организации связей в состав атрибутов сущности вводится искусственный ключ (обычно число). Ключевой атрибут (набор атрибутов) на диаграмме отмечается двумя линиями снизу, внешние ключи отмечаются одной линией. В табл. 1.1 представлены основные элементы, используемые для формирования ER-диаграммы в нотации Чена.

Таблица 1.1 – Основные элементы ER-диаграммы в нотации Чена

Элемент диаграммы	Описание
	Независимая сущность
	Зависимая сущность
	Связь
	Идентифицирующая связь
	Атрибут
	первичный ключ
	внешний ключ

Связь имеет кардинальное число, определяющее, какое количество экземпляров одной сущности имеет информационную связь с экземплярами другой сущности.

На рис 1.1 представлен пример *ER*-диаграммы в нотации Чена.

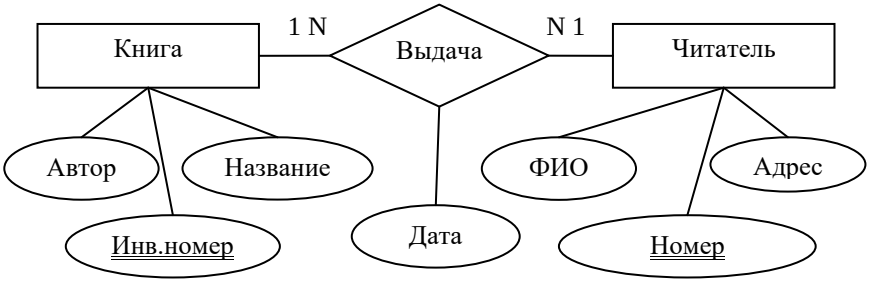


Рисунок 1.1 – Диаграмма сущность-связь в нотации Чена

1.2.3. Нотация IDEF1X.

В нотации *IDEF1X* все сущности делятся на зависимые и независимые от идентификаторов аналогично нотации Чена. Независимая сущность изображается в виде обычного прямоугольника, зависимая – в виде прямоугольника с закругленными углами (табл. 1.2.).

Таблица 1.2 – Обозначение сущностей нотации IDEF1X

Элемент диаграммы	Тип сущности
имя	независимая сущность
имя	зависимая сущность

Список атрибутов приводится внутри прямоугольника, обозначающего сущность. Атрибуты, составляющие ключ сущности, группируются в верхней части прямоугольника и отделяются горизонтальной чертой.

Связь изображается линией, проводимой между сущностью-родителем и сущностью-потомком, с точкой на конце линии у сущности-потомка. Если экземпляр сущности-потомка однозначно определяется своей связью с сущностью-родителем, то связь называется идентифицирующей, в противном случае — неидентифицирующей. Идентифицирующая связь изобража-

ется сплошной линией, неидентифицирующая – пунктирной линией. В табл. 1.3 приведено обозначение связей в нотации *IDEF1X*.

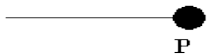
Таблица 1.3 – Типы связей нотации *IDEF1X*

Элемент диаграммы	Описание
_____	идентифицирующая связь
– – – – – – – – – – – – –	неидентифицирующая связь

В отличие от нотации Чена связи не имеют самостоятельных атрибутов. При необходимости отобразить связь с атрибутами (характеристиками) она моделируется как сущность.

Связи имеют кардинальное число (мощность связи). По умолчанию мощность связи принимается равной *M* (много). Обозначение кардинальности связей представлено в табл. 1.4.

Таблица 1.4 – Обозначение кардинального числа

Элемент диаграммы	Описание
_____	1,1 – связь один к одному
	0,M – связь один ко многим необязательная
	0,1 – связь один к одному необязательная
	1,M – связь один ко многим обязательная
	точно *N* связей (*N* - произвольное число)

Жирная точка ставится для подчиненной сущности. В *IDEF1X* существуют следующие виды мощностей связей:

- N – каждый экземпляр сущности-родителя может иметь ноль, один или более одного связанного с ним экземпляра сущности-потомка (по умолчанию);

- P – каждый экземпляр сущности-родителя должен иметь не менее одного связанного с ним экземпляра сущности-потомка;
- Z – каждый экземпляр сущности-родителя должен иметь не более одного связанного с ним экземпляра сущности-потомка;
- конкретное число – каждый экземпляр сущности-родителя связан с некоторым фиксированным числом экземпляров сущности-потомка.

На рис. 1.2 представлен пример диаграммы в нотации IDEF1X.

Рисунок 1.2 – Диаграмма сущность-связь в нотации IDEF1X

1.2.4. Нотация Мартина

В нотации Мартина также вводится понятие зависимой и независимой сущностей. Кроме того, сущности могут иметь иерархическую связь «родитель-потомок».

Список атрибутов приводится внутри прямоугольника, обозначающего сущность. Ключевые атрибуты подчеркиваются. Связи изображаются линиями, соединяющими сущности, вид линии в месте соединения с сущностью определяет кардинальность связи (табл.1.5)

Таблица 1.5 – Обозначение связей в нотации Мартина

Обозначение	Кардинальность
	кардинальное число не определено
	один к одному
	один к одному, необязательная

Продолжение таблицы 1.5

Обозначение	Кардинальность
	один ко многим
	многие ко многим, необязательная
	многие ко многим, обязательная

Имя связи указывается на линии, её обозначающей. Пример *ER*-диаграммы представлен на рис. 1.3.

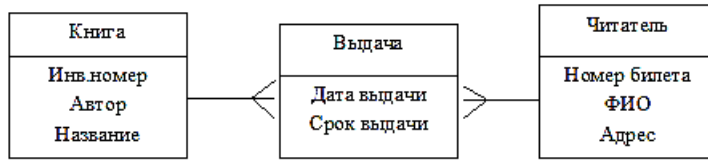


Рисунок 1.3 – Диаграмма сущность-связь в нотации Мартина

1.2.5. Нотация Баркера

Сущности обозначаются прямоугольниками, внутри которых приводится список атрибутов. Ключевые атрибуты отмечаются символом # (решетка).

Связи обозначаются линиями с именами, место соединения связи и сущности определяет кардинальность связи.

Пример *ER*-диаграммы представлен на рис. 1.4.

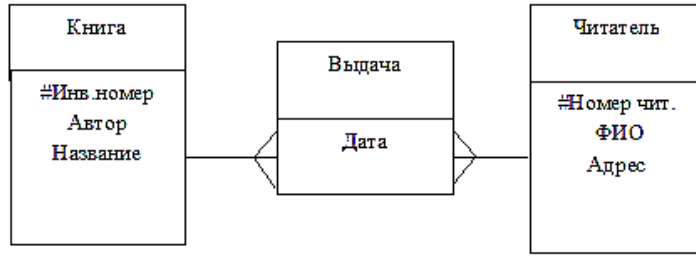


Рисунок 1.4 – Диаграмма сущность-связь в нотации Беркера

1.3. Работа с CASE-приложением разработки схемы базы данных

Приложение Oracle SQL Developer Data Modeler обеспечивает разработку схемы базы данных с применением *ER*-метода.

Приложение позволяет проектировать логическую и/или реляционную структуру базы данных (рис.1.5). Если при разработке схемы базы данных неизвестна СУБД, которая будет использоваться для хранения информации, осуществляется проектирование на логическом уровне. Схема реляционного уровня может быть автоматически преобразована в физическую для соответствующей СУБД

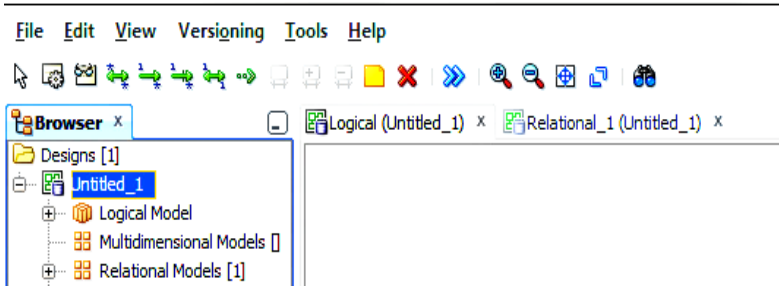


Рисунок 1.5 – Основное окно приложения разработки схемы база данных

Приложение поддерживает несколько видов нотации для обозначения сущностей и связей: нотация Баркера, нотация Бахмана, нотация IE (Information Engenering). Нотация Чена не поддерживается. Это приводит к тому что, если связь имеет самостоятельные атрибуты в Oracle SQL Developer Data Modeler, то такая связь моделируется как сущность.

Переключение вида нотации осуществляется с помощью пункта меню View->Logical Diagram Notation (рис.1.6).

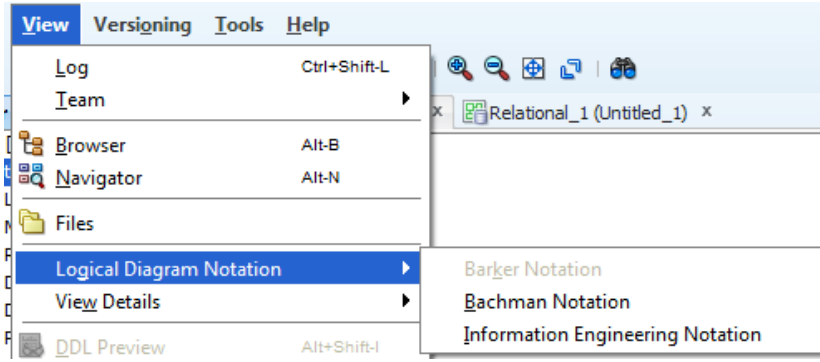


Рисунок 1.6 –Выбор нотации при проектировании схемы база данных

Разработка схемы базы данных может осуществляться с помощью элементов сущность (Entity) и связь (Relation), которые находятся в блоке средств проектирования.

Набор средств проектирования зависит от вида модели. Для логической модели используются сущность, представление и связи (рис. 1.7), для реляционной модели – таблица, представление, внешний ключ (рис. 1.8).

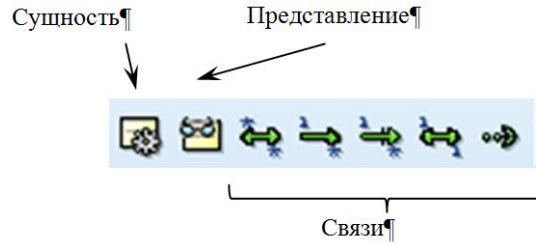


Рисунок 1.7 – Элементы проектирования логической схемы база данных

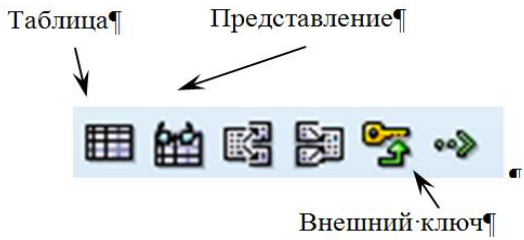


Рисунок 1.8 – Элементы проектирования реляционной схемы база данных

1.3.1. Построение логической структуры базы данных

Добавление сущностей, выделенных в предметной области, к схеме осуществляется переносом элемента Entity из блока инструментов в область логической схемы (рис. 1.9). По мере формирования набора сущностей могут добавляться связи между сущностями. Для создания связи между сущностями после выбора типа связи из набора инструментов (многие ко многим, один ко многим или один к одному) первоначально указывается основная таблица (со стороны «один»), а затем дополнительная (со стороны «многие»).

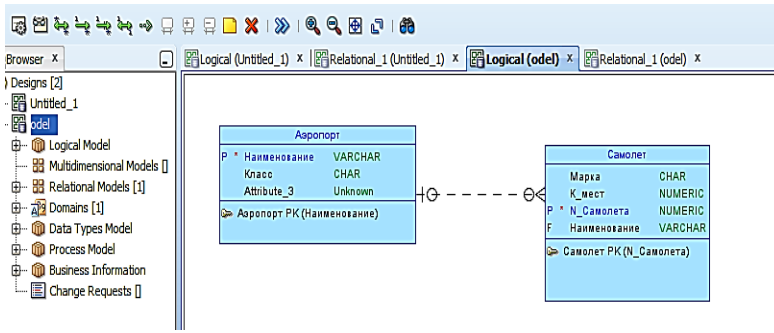


Рисунок 1.9 – Формирование логической схемы базы данных

Добавление атрибутов сущностей осуществляется через контекстное меню и пункт Properties.

Для каждого атрибута вводится имя и выбирается тип. Тип атрибута может быть выбран из списка логических типов (Logical) или из списка пользовательских типов (Domain) (рис.1.10). При формировании логической структуры базы данных можно задавать для выбранных типов размер (количество символов) и для числовых дополнительно масштаб.

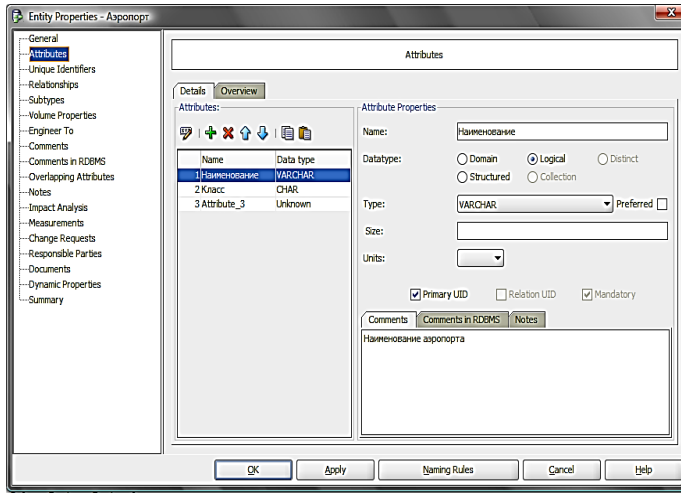


Рисунок 1.10 – Формирование атрибутов сущности

1.3.2. Построение реляционной модели схемы базы данных

Реляционная структура базы данных может быть получена из логической или может быть сформирована непосредственно разработчиком с помощью инструментов построения реляционной схемы (см. рис. 1.8).

Для автоматического построения реляционной схемы базы данных необходимо выбрать элемент «»» из панели инструментов. С помощью диалогового окна (рис.1.11) можно выбрать сущности, которые будут участвовать в построении реляционной схемы.

Отличие реляционной схемы базы данных от логической состоит в том, что к сущностям будут добавлены поля с внешними ключами (рис.1.12).

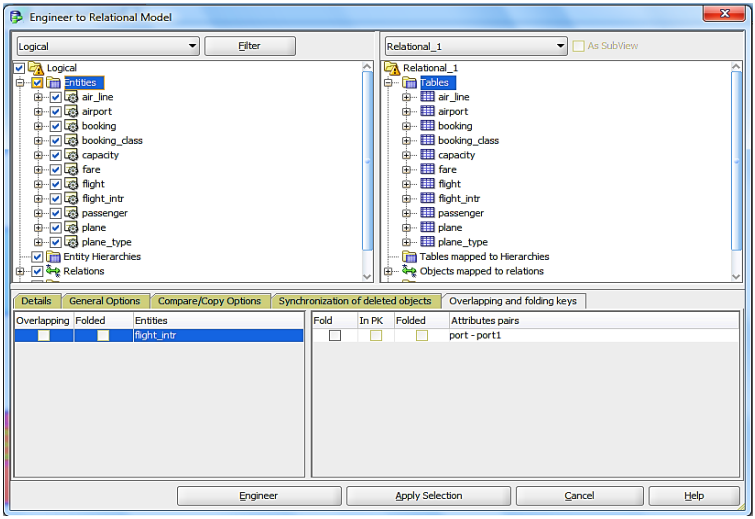


Рисунок 1.11 –Диалоговое окно генерации реляционной модели

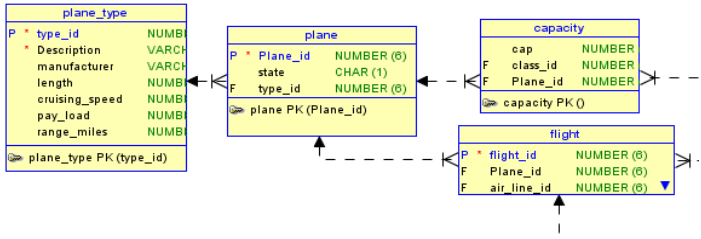


Рисунок 1.12 –Реляционная модель базы данных

1.3.3. Построение физической модели базы данных

Физическая модель базы данных представляет собой набор команд SQL для формирования таблиц и представлений. Физическая модель привязана к конкретной СУБД.

Физическую модель формируют из реляционной модели. В процессе создания физической модели логические типы данных полей сущностей будут преобразованы в типы данных, поддерживаемые СУБД. Каждая сущность сформирует отдельную таблицу.

Для построения физической схемы выбирается элемент средств проектирования. В диалоговом окне создания физической схемы можно выбрать тип СУБД. В результате генерации будут сформированы команды SQL (рис. 1.13).

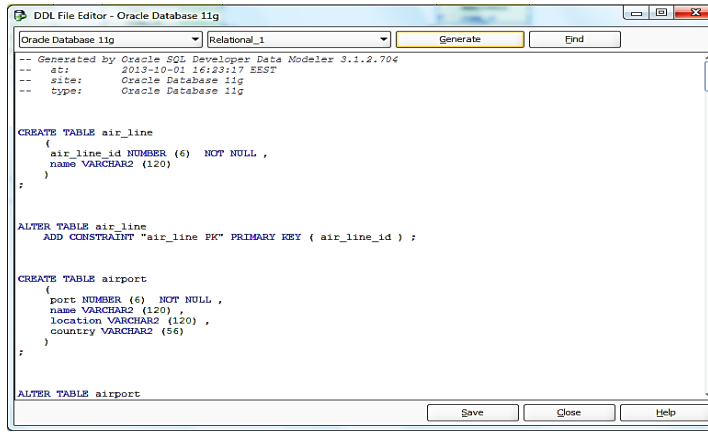


Рисунок 1.13 – Окно генерации команд SQL

1.4. Задание для выполнения лабораторной работы

1.4.1. Порядок выполнения

1. Для предметной области разработать набор сущностей, информация о которых должна храниться в базе, и совокупность связей между сущностями с учетом ограничений.

2. Для каждой сущности задать набор атрибутов и их типы. Определить набор первичных и потенциальных ключей, добавить, при необходимости, искусственные ключи.

3. Установить связи между сущностями в соответствии с типом. Задать имена связей и определить кардинальное число.

4. Построить реляционную модель базы данных.

5. Получить схему базы данных для выбранной СУБД и сформировать команды создания таблиц и индексов.

6. Выполнить сгенерированные команды SQL для формирования таблиц и индексов БД.

7. Оформить отчет о проекте БД.

1.4.2. Варианты заданий

1. База данных "Железная дорога" хранит информацию:

вагоны – марка вагона, вместимость/грузоподъемность, стоимость;

поезд – набор вагонов, шифр поезда;

дорога –наименование, протяжённость, число колей, средняя стоимость эксплуатации;

станция – наименование, число путей.

Каждый вагон имеет свой номер и дату ввода в эксплуатацию.

Поезд состоит из нескольких вагонов, возможно, разных типов. Общее число вагонов не должно превышать 20. Поезд движется через несколько станций.

Дорога проходит через несколько станций. При этом одна и та же станция может принадлежать различным дорогам (узловая станция).

2. База данных "Парки города" хранит информацию:

парки – наименование, площадь, плотность посадки, место нахождения (адрес);

насаждения – тип культуры, наименование, средняя продолжительность жизни;

фонтаны – шифр, дата постройки, расход воды (максимальный и нормальный), площадь;

павильоны – наименование, тип (кафе, продуктовый, развлекательный, прокат вещей), занимаемая площадь.

Каждый парк имеет собственное имя. В парке высажены определённые насаждения. База данных должна хранить информацию о количестве насаждений каждого типа.

В парке могут находиться фонтаны и павильоны.

3. База данных "Поликлиника"хранит информацию:

врач – фамилия, имя, отчество, специальность, дата устройства на работу;

посетитель – фамилия, имя, отчество, домашний адрес;

прием – часы приёма (начало приёма, окончание приёма), номер дня недели, номер кабинета;

процедурный кабинет –время начала и время окончания работы, номер дня недели, номер кабинета, название лаборатории.

Каждый врач может иметь несколько специальностей и работать в поликлинике не более чем на двух специальностях (совместитель). Каждый врач осуществляет приём в определенные часы в одном из кабинетов.

Посетитель может записаться к врачу на определенное время или в процедурный кабинет.

4. База данных " Оптовая база " хранит информацию:

товар – наименование, стоимость, количество, дата поставки;

поставщик – наименование фирмы, телефон, адрес, фамилия директора;

потребитель – телефон, адрес, фамилия;

поставка – наименование товара, количество, дата поставки;

заказ – наименование товара, количество, дата заказа, дата выполнения заказа.

Товар, хранящийся на складе, может иметь различную стоимость в зависимости от поставщика и даты поставки. Каждая поставка связана только с одним товаром конкретного поставщика.

Потребители могут заказывать различные товары. Один заказ может оформляться на несколько товаров. При этом, если товар отсутствует на складе, дата выполнения может отсутствовать в заказе.

5. База данных " Библиотека " хранит информацию:

книга – название, авторы, стоимость, дата поступления, инвентарный номер;

читатель – фамилия, имя отчество, дата рождения, домашний адрес;

автор – фамилия, имя, отчество, дата рождения, дата смерти, краткая биография;

издательство – наименование, телефон отдела заказов, адрес;

Каждая книга имеет свой инвентарный номер. В библиотеке могут храниться несколько книг одного названия и автора, но разного издательства и даты поступления. Книга может иметь несколько авторов.

Читателю могут выдаваться несколько книг, но не более пяти. При этом срок выдачи не может превышать один месяц.

6. База данных " Автопарк" хранит информацию:

легковой автомобиль – тип автомобиля, число пассажиров, мощность двигателя, расход топлива;

грузовой автомобиль – тип автомобиля, число осей, грузоподъемность, объем кузова, мощность двигателя, расход топлива;

работник – фамилия, имя отчество, дата рождения, домашний адрес, специальность.

Каждый грузовой и легковой автомобиль имеет индивидуальный государственный номер. За каждым автомобилем закреплен один водитель и несколько мастеров для обслуживания.

В автопарке имеются несколько автомобилей одного типа.

Контрольные вопросы

1. Какие группы методов проектирования схемы базы данных существуют в настоящее время?
2. На чем основаны методы проектирования схемы базы данных с использованием ER-диаграмм?
3. Какие достоинства и недостатки методов ER-диаграмм проектирования схемы баз данных?
4. На чем основаны методы проектирования с использованием функциональных и многозначных зависимостей?
5. Какие достоинства и недостатки методов проектирования с использованием функциональных и многозначных зависимостей?
6. Что понимается под термином сущности в методах ER-диаграмм?
7. Что должна иметь каждая сущность?
8. Что такое атрибут сущности?
9. Какие типы сущностей и связей различаются в нотации Чена?
10. Что определяет кардинальное число?
11. Какие существуют нотации для описания модели данных?
12. Чем отличается нотация IDEF1X от нотации Чена?

13. Какова технология проектирования схемы базы данных при использовании приложения Oracle SQL Developer Data Modeler?

14. С помощью каких элементов осуществляется построение логической схемы базы данных?

15. Как формируется реляционная схема базы данных?

16. Что требуется для преобразования реляционной схемы базы данных в физическую?

2. ЛАБОРАТОРНАЯ РАБОТА 2
Построение базы данных на основе функциональных и многозначных зависимостей

***Цель работы*:** Изучение алгоритмов построения логической модели базы данных на основе функциональных и многозначных зависимостей. Получение навыков приведения схемы базы данных к нормальным формам.

2.1. Проектирование схемы базы данных методами функциональных и многозначных зависимостей

Задача проектирования реляционной базы данных заключается в выборе схемы базы из множества альтернативных вариантов, т. е., по сути, требуется определить набор схем отношений базы данных. Для удовлетворения этих требований необходимо определить, из каких отношений должна состоять база данных и какие атрибуты будут входить в эти отношения.

Проектирование на основе функциональных и многозначных зависимостей заключается в том, что предполагается существование некоторого универсального отношения, содержащего все атрибуты базы данных. На основе анализа связей между атрибутами осуществляется декомпозиция универсального отношения – переход к нескольким отношениям меньшей размерности. Новые отношения удовлетворяют определенным условиям, и

при этом исходное отношение должно восстанавливаться с помощью операции естественного соединения.

Функциональные (ФЗ) и многозначные (МЗ) зависимости между значениями атрибутов схемы базы данных определяются на основе семантического анализа предметной области.

2.1.1. Функциональные зависимости и их свойства

Если $R = (U)$ – схема отношения, $U = \{A1, A2, ..., An\}$ – множество атрибутов и $X, Y \subseteq U$, то X функционально определяет Y, если в отношении R не могут содержаться два кортежа, компоненты которых совпадают по всем атрибутам, принадлежащим множеству X, но не совпадают хотя бы по одному атрибуту, принадлежащему множеству Y.

Функциональные зависимости между множествами атрибутов $X, Y \subseteq U$ обычно обозначаются как $X \rightarrow Y$.

Множество функциональных зависимостей между атрибутами отношения называется системой образующих структуры функциональных зависимостей отношения в целом и обозначается как $FD(R(U))=\{ X \rightarrow Y \mid X, Y \subseteq U \}$.

Множество всех возможных функциональных зависимостей называется замыканием функциональных зависимостей FD^+. Если $FD^+ = FD$, то FD называют замкнутым множеством зависимостей.

Для построения системы функциональных зависимостей $FD(R(U))$ отношения требуется, в общем случае, перебрать все возможные подмножества атрибутов $X \subseteq U$ для проверки функциональной зависимости $X \rightarrow Y$ при $Y \subseteq U$ и $Y \cup X = \emptyset$. Кроме того, проверка наличия функциональной зависимости может проводиться только при заполнении отношения всеми возможными данными.

Поскольку число подмножеств множества атрибутов U определяется как 2^n (n – количество атрибутов множества U) и заполнение отношения всеми возможными данными либо невозможно, либо требует больших затрат, практическим способом выделения функциональных зависимостей является анализ семантики (смыслового значения) атрибутов.

Например, если отношение содержит сведения о работниках предприятия: фамилия, имя, отчество, идентификационный код и домашний адрес, то могут быть выделены следующие функциональные зависимости:

идентификационный код → фамилия

идентификационный код → имя

идентификационный код → отчество

идентификационный код →домашний адрес,

так как идентификационный код присваивается отдельному работнику и не существует другого работника с таким же идентификационным кодом.

Однако если работник изменил фамилию и отношение должно хранить сведения о старой и новой фамилии в виде одного атрибута, то функциональная зависимость между идентификационным кодом и фамилией отсутствует.

Некоторые функциональные зависимости могут быть получены на основе других функциональных зависимостей с помощью набора аксиом Армстронга:

1) если $X \supseteq Y$, то $X \rightarrow Y$ (рефлективность);

2) если $X \rightarrow Y$ и $W \supseteq Z$, то $X \cup W \rightarrow Y \cup Z$ (продолжение);

3) если $X \rightarrow Y$ и $Y \rightarrow Z$, то $X \rightarrow Z$ (транзитивность).

Легко видеть, что аксиомы 2) и 3) могут быть объединены в одну аксиому:

4) если $X \rightarrow Y$ и $Y \cup W \rightarrow Z$, то $X \cup W \rightarrow Z$ (псевдотранзитивность).

Полезны также следующие свойства, вытекающие из 1)–3):

5) если $X \rightarrow Y$ и $X \rightarrow Z$, то $X \rightarrow Y \cup Z$ (аддитивность);

6) если $X \rightarrow Y$ и $Z \subseteq Y$, то $X \rightarrow Z$ (декомпозиция).

Аксиомы Армстронга можно рассматривать как правила вывода, позволяющие выводить функциональные зависимости, логически следующие из заданного набора зависимостей.

2.1.2. Базис структуры функциональных зависимостей

Из множества FD^+ можно выделить подмножество зависимостей F_B, которое имеет свойство:

любая функциональная зависимость $f \in FD^+$ может быть получена из функциональных зависимостей F_B путем применения аксиом Армстронга и при этом из F_B нельзя удалить ни одну функциональную зависимость без потери возможности получить FD^+.

Множество F_B в этом случае называется базисом структуры функциональных зависимостей FD^+. Для отношения R, имеющего замыкание FD^+, может иметься несколько различных базисов F_{Bi}.

2.1.3. Нормальные формы отношений

Нормализация отношений – пошаговый процесс разложения (декомпозиции) исходных отношений базы данных на более простые. В настоящее время выделяют шесть основных форм. Каждая следующая нормальная форма (НФ) имеет «лучшие» свойства, чем предыдущая. Каждой нормальной форме соответствует некоторый набор ограничений. Отношение находится в определенной нормальной форме, если оно удовлетворяет набору ограничений этой формы:

• 1НФ – значения всех атрибутов в них атомарные;

• 2НФ – отношение находится в 1НФ, и в нём отсутствуют функциональные зависимости атрибутов от части ключа;

• 3НФ – отношение находится во 2НФ и в нём отсутствуют транзитивные зависимости между неключевыми атрибутами;

• НФБК (нормальная форма Бойса–Кодда) – отношение находится в 3НФ, и никакие атрибуты в нём не зависят транзитивно ни от одного ключа отношения;

• 4НФ – отношение находится в НФБК и содержат только зависимости от потенциальных ключей (отсутствует многозначная зависимость);

• 5НФ – отношение находится в 4НФ, и в нём отсутствуют любые зависимости соединения.

2.1.4. Построение отношений в третьей нормальной форме.

Построение отношений в первой и второй нормальных формах не требует больших затрат. Атомарность атрибутов обеспечивается возможностью хранения только одного значения в каждый момент времени в атрибутах каждого кортежа. Во второй нормальной форме любые два кортежа

должны отличаться хотя бы в значении одного атрибута, т.е. нет повторяющихся кортежей.

Базис структуры функциональных зависимостей позволяет построить схему базы данных в третьей нормальной форме. Основой алгоритма формирования третьей нормальной формы является теорема Хита.

Теорема Хита. Пусть задана схема отношения $R = (U, F)$, в котором $U = X \cup Y \cup Z$,а зависимость $X \rightarrow Y \in F$., тогда декомпозиция отношения $R = (R_1, R_2)$, где $R_1 = (X \cup Y)$, $R_2 = (X \cup Z)$, обладает свойством соединения без потерь.

Согласно теореме Хита, отношение $R(U)$, имеющее базис F_B , может быть представлено в виде набора отношений $R_i(X_i \cup Y_i)$, если $X_i \rightarrow Y_i \in F_B$.

2.1.5. Многозначная зависимость атрибутов

Если функциональная зависимость определяет зависимые значения атрибутов, то многозначная зависимость строится на основе независимости значений.

Пусть S – схема отношения R; A и B – непересекающиеся подмножества U, и пусть $C = U \setminus (A \cup B)$. Отношение R со схемой S удовлетворяет многозначной зависимости $A \rightarrow\rightarrow B$, если при заданных значениях из A для двух произвольных кортежей $t_1(A) = t_2(A)$ в R существует кортеж t_3, для которого выполнены равенства

$$t_3(A) = t_1(A),\ t_3(B) = t_1(B) \text{ и } t_3(C) = t_2(C)$$

т.е. если два кортежа совпадают по атрибутам A, найдётся третий кортеж, который будет совпадать по атрибутам A и B с первым кортежем и по атрибутам C со вторым кортежем.

Например, если отношение содержит атрибуты: «лектор», «группа», «предмет», то непосредственной зависимости между атрибутами «группа» и «предмет» может не быть. Однако если каждый лектор проводит занятия в каждой группе по соответствующим предметам, то отношение имеет две зависимости типа один-ко-многим – «лектор»-«группа» и «лектор»-«предмет».

Наличие этих зависимостей позволяет разбить основное отношение на два других: {«лектор», «группа»} и {«лектор», «предмет»}.

Известно, что для данного отношения R со схемой S, содержащей множество атрибутов $\{A, B, C\}$, многозначная зависимость $A \rightarrow\!\!\rightarrow B$ выполняется тогда и только тогда, когда также выполняется многозначная зависимость $A \rightarrow\!\!\rightarrow C$. Многозначные зависимости всегда образуют связанные пары, и поэтому их обычно представляют вместе в символическом виде:

$$A \rightarrow\!\!\rightarrow B \mid C.$$

К многозначным зависимостям применим следующий набор аксиом:

1) если $X \cup Y \cup Z = U$ (U—множество всех атрибутов отношения R), $Y \cap Z \subseteq X$, то при $X \rightarrow\!\!\rightarrow Y$ имеет место $X \rightarrow\!\!\rightarrow Z$ (дополнение);

2) если $X \supseteq Y$, то $X \rightarrow\!\!\rightarrow Y$, (рефлективность);

3) если $X \rightarrow\!\!\rightarrow Y$, и $W \subseteq Z$, то $X \cup Z \rightarrow\!\!\rightarrow Y \cup W$ (присоединение);

4) если $X \rightarrow\!\!\rightarrow Y$ и $Y \rightarrow\!\!\rightarrow Z$, то $X \rightarrow\!\!\rightarrow Z \setminus Y$ (транзитивность);

5) если $X \rightarrow\!\!\rightarrow Y$ и $Y \cup W \rightarrow\!\!\rightarrow Z$, то $X \cup W \rightarrow\!\!\rightarrow Z \setminus (Y \cup W)$ (псевдотранзитивность);

6) если $X \rightarrow\!\!\rightarrow Y$ и $X \rightarrow\!\!\rightarrow Z$, то $X \rightarrow\!\!\rightarrow Y \cup Z$ (аддитивность);

7) если $X \rightarrow\!\!\rightarrow Y_1$ и $X \rightarrow\!\!\rightarrow Y_2$, то $X \rightarrow\!\!\rightarrow Y_1 \cap Y_2$, $X \rightarrow\!\!\rightarrow Y_1 \setminus Y_2, X \rightarrow\!\!\rightarrow Y_2 \setminus Y_1$ (декомпозиция).

Использование многозначной зависимости позволяет декомпозировать отношение R на два других – R_1 и R_2 и получить четвертую нормальную форму.

Теорема Фейджина. Пусть A, B и C являются множествами атрибутов отношения R. Отношение R будет равно соединению его проекций $\{A, B\}$ и $\{A, C\}$ тогда и только тогда, когда для отношения R выполняется многозначная зависимость $A \rightarrow\!\!\rightarrow B \mid C$.

2.2. Проектирование схемы базы данных

В настоящее время известно несколько алгоритмов проектирования схемы базы данных на основе функциональных и многозначных зависимостей. Основные недостатки этих алгоритмов – сложность построения базиса

зависимостей и возможное формирование отношений в неестественной (не соответствующей представлению разработчика и пользователя базы данных) форме.

2.2.1. Построение базиса функциональной зависимости

Формирование базиса структуры функциональных зависимостей можно осуществить на основе построения замыкания атрибутов.

Пусть задана схема $R = (U, F)$ и $X \subset U$, требуется вычислить X^+.

Шаг 1: положить $X(0) := \emptyset$, $i := 0$;

Шаг 2: найти $\Delta X(i+1)$ – множество всех таких атрибутов $y \in U \setminus X(i)$, что существует некоторая зависимость $V \rightarrow W \in F$, такая что $X(i) \supseteq V$ и $y \in W$;

Шаг 3: если $\Delta X(i+1) = \emptyset$, то $X(i) = X^+$, конец работы,
иначе, $X(i+1) := X(i) \cup \Delta X(i+1)$, $i := i + 1$ и перейти к шагу 2.

2.2.2. Алгоритм Делобеля – Кейси

Пусть задана схема отношений $R = (U, F)$.

1. Перейти от F к элементарному функциональному базису F_B, т. е. к минимальному покрытию, состоящему только из полных зависимостей, причем в правой части каждой зависимости должен находиться только один атрибут.

2. По каждой зависимости $(X_i \rightarrow Y_i) \in F_B$ образовать подсхему $R_i = (U_i = X_i \cup Y_i)$; если при этом для некоторых i и j, $i \neq j$ окажется, что $U_j \subseteq U_i$, то R_j удаляется.

3. Если хотя бы одна из полученных подсхем R_i содержит ключ исходного отношения R, то совокупность полученных R_i $i=1,n$ является 3НФ отношения R, иначе добавить к полученной декомпозиции еще одну схему R_{n+1}, состоящую из ключей отношений R_i.

2.2.3. Алгоритм Бернштейна

1. Устранить в функциональных зависимостях из G избыточные атрибуты, получая в результате набор G'.

2. Найти базис F_B замыкания набора G'.

3. Разбить F_B на группы H_i так, чтобы все функциональные зависимости в одной группе имели идентичные левые стороны.

4. Образовать множество $J = \varnothing$. Для каждой пары групп H_i и $H_j \in$ FB, у которых левыми сторонами являются соответственно X и Y, слить вместе H_i и H_j, если в F_B входит биекция $X \leftrightarrow Y$, добавить $X \to Y$ и $Y \to X$ к J. Для каждого атрибута $A \in Y$ удалить из F_B зависимость $X \to A$, если она уже входит в J. Аналогичные действия проделать с зависимостями $Y \to B \in F_B$, у которых $B \in X$.

5. Устранить в H транзитивные зависимости, которые могут появиться после выполнения шага 4. Добавить все функциональные зависимости из J в соответствующую группу набора F_B.

6. Для каждой группы конструировать отношение, состоящее из всех атрибутов этой группы. Каждое множество атрибутов левой стороны какой-либо функциональной зависимости из данной группы является ключом отношения.

2.2.4. Алгоритм Ислура

1. Устранить в функциональных зависимостях из G избыточные атрибуты, получив в результате набор G'.

2. Разделить G' на группы так, чтобы все функциональные зависимости в одной группе имели идентичные левые стороны. Зависимости одной группы объединить в одну функциональную зависимость. Полученному множеству функциональных зависимостей дать имя H.

3. Построить расширение $H^{\oplus}$.

4. Выявить биекции и эквивалентные ключи. Для этого из каждого подмножества функциональных зависимостей с левыми сторонами $L_{j1}, L_{j2}, \ldots, L_{jr}$ такого, что $L_{j1} \bigcup T_{j1} = L_{j2} \bigcup T_{j2} = \ldots L_{jr} \bigcup T_{jr}$ удалить функциональные зависимости $\{L_{jk} \to T_{jk}\}$ для $2 \le k \le r$ из $H^{\oplus}$. Образовать множества $\{L_{j1}, L_{j2}, \ldots, L_{jr}\}$ эквивалентных ключей. Обозначим H' – новое множество функциональных зависимостей.

5. Получить транзитивную редукцию множества H'.

6. Для каждой зависимости конструировать отношение, состоящее из всех атрибутов левой и правой сторон. Множество атрибутов левой стороны является ключом отношения. Включить все эквивалентные ключи в од-

но отношение.

2.2.5. Алгоритм Неклюдовой–Цаленко

Алгоритм Неклюдовой-Цаленко состоит из следующих шагов:

1. Найти все возможные ключи и построить множество первичных атрибутов U_p.

2. В множестве G найти такое минимальное подмножество G', что $\{G' \cup FD(U_p)\}^+ = G^+ = FD(U)$.

3. Устранить в функциональных зависимостях из G' избыточные атрибуты.

4. Исключить из G' все функциональные зависимости вида $K \rightarrow A$, где K – возможный ключ. Построить множество U_1, добавив к U_p элементы A, входящие в правые части исключенных из G' зависимостей. Если $U_1 = U$, то U в 3НФ и работа алгоритма закончена. Если $U_1 \neq U$, то перейти к шагу 5.

5. Обозначить G'' – множество функциональных зависимостей, оставшихся в G'. Положить $U' = \{A \mid A$ входит в зависимость из $G''\}$. Если $U' \neq U$, то найти систему образующих для полной подструктуры $FD(U')$ и повторить шаги 1—4 для U'. Синтаксическое разложение для U состоит из синтаксического разложения для U' и множества U_1. Если $U' = U$, то применить алгоритм Бернштейна и построить разложение для U. Если в построенном разложении хотя бы одно слагаемое содержит возможный ключ, то разложение является синтаксическим.

2.3. Задание для выполнения лабораторной работы

2.3.1. Порядок выполнения

1. Для предметной области разработать набор атрибутов универсальной таблицы.

2. Выявить набор функциональных и многозначных зависимостей между атрибутами.

3. Используя алгоритм Бернштейна разработать набор таблиц базы данных. Проверить полученные таблицы на соответствие третьей и четвертой нормальным формам.

4. Установить связи между таблицами и построить реляционную модель базы данных.

5. Получить схему базы данных для выбранной СУБД и сформировать команды создания таблиц и индексов.

6. Выполнить сгенерированные команды SQL для формирования таблиц и индексов БД.

7. Оформить отчет о проекте БД.

2.3.2. Варианты заданий

1. База данных "Кино и кинотеатры" хранит информацию о фильмах (наименование фильма, год выпуска в прокат, фамилия режиссёра, список актеров) и кинотеатрах (наименование, число залов, количество мест в залах, сеансы, количество проданных мест на соответствующие сеансы, репертуар на месяц).

2. База данных "Автомобили" хранит информацию об автомобилях (серия автомобиля, завод изготовитель, мощность двигателя, число пассажиров, вес автомобиля, цвет, тип кузова, наличие кондиционера) и сервисных центрах обслуживания автомобилей (наименование фирмы, наименование сервисного центра, число мест для ремонта и обслуживания автомобилей, адрес центра).

3. База данных "Цветы" хранит информацию о растениях (наименование, требуемая почва, период цветения, форма цветка, цвет, размер цветка, сроки хранения) и о магазинах продажи (адрес магазина, наименование магазина, ассортимент, площадь торгового зала, число работников, вид торговли – оптовая или розничная, наличие заказов).

4. База данных " Отдел кадров " хранит информацию о работниках предприятия (фамилия имя отчество работника, должность, специальность, разряд по специальности, дата приема на работу, дата увольнения с работы, совместитель или основной работник, паспортные данные) и подразделениях (наименование подразделения, число работников).

5. База данных "Аквариум" хранит информацию об аквариумных рыбках (максимальный размер в сантиметрах, требуемая температура воды, pH воды, требуемый объем аквариума в литрах, необходимость укрытий), растениях для аквариумов (вид – кустовое, донное, плавающее, длина листа в

сантиметрах), заказах на аквариум (состав растений и набор рыбок, дата заказа заказчик).

6. База данных "Аэропорт" хранит информацию о самолетах (бортовой номер, число посадочных мест, дальность перелёта, номер рейса, стоимость билетов по классам) и пассажирах (фамилия, имя, отчество, пункт отправления и пункт назначения, дата отправления, номер самолёта).

Контрольные вопросы

1. В чем состоит основа методов проектирования схемы базы данных с использованием функциональных и многозначных зависимостей?
2. Как выделяются функциональные зависимости между атрибутами отношений?
3. Как выделяются многозначные зависимости атрибутов отношений?
4. Что понимается под системой образующих структуры функциональных зависимостей отношения?
5. Что такое базис функциональных зависимостей?
6. Для чего используется нормализация отношений?
7. Какие существуют нормальные формы отношений?
8. Что такое атрибут отношения?
9. Как формулируется теорема Хита?
10. Как определяется многозначная зависимость атрибутов отношения?
11. Какие аксиомы применимы к многозначным зависимостям?
12. Как формулируется теорема Фейджина?
13. Как можно построить базис функциональных зависимостей?
14. Из каких шагов состоит алгоритм Делобеля–Кейси проектирования схемы базы данных?
15. В чем отличие алгоритма Бернштейна от алгоритма Делобеля–Кейси?
16. В чем состоит особенность алгоритма Неклюдовой–Цаленко при проектировании схемы базы данных?

Список литературы

1. Дейт К. Введение в системы баз данных/ К. Дейт.– 7-е издание – М.: Вильямс, 2001. – 1072 с.

2. Дейт К. Дж. Руководство по реляционной СУБД DB2/ К. Дж. Дейт: пер. с англ. – М.: Финансы и статистика, 1988. – 320 с.

3. Диго С.М. Проектирование баз данных : учебник для вузов / С.М. Диго.– М.: Финансы и статистика, 1988. – 216 с.

4. Евдокимов В.В. и др. Экономическая информатика : учебник для вузов / Под ред. проф. Евдокимова. – СПб.: Питер, 1997. – 592 с.

5. Когаловский М.Р. Абстракции и модели в системах баз данных / М.Р. Когаловский.//Системы управления базами данных – 1998. – № 4–5

6. Мартин Дж. Организация баз данных в вычислительных системах/ Дж. Мартин: пер. с англ. – М.: Мир, 1980.–622 с.

7. Конолли Т. Базы данных: проектирование, реализация и сопровождение. Теория и практика / Т. Конолли, К. Бегг, А. Страчан. 2-е издание. – М.: Вильямс, 2000. – 1120 с.

Содержание

Навчальне видання

Методичні вказівки
до лабораторних робіт
«Методи проектування схем баз даних»
з дисципліни
«Розподілені інформаційно-аналітичні системи»

Російською мовою
Укладачі:
КОЖИН Юрій Миколайович
МАЛИХ Олег Миколайович
ПРОКОПЕНКОВ Володимир Пилипович

Відповідальний за випуск *О.С.Куценко*

Роботу до друку рекомендував *О.В.Горілий*
Редактор *О.С. Самініна*

План 2016, поз. 87

Підп. до друку 23.03.2017	Формат 60х84 1/16.	Папір офсетний.
Riso–друк.	Гарнітура Таймс.	Ум. друк. арк. 1,8.
Наклад 25 прим.	Зам. №	Ціна договірна.

Видавничий центр НТУ “ХПІ”.
вул. Кирпичова, 21, м. Харків, 61002
Свідоцтво суб’єкта видавничої справи ДК №3657 від 24.12.2009 р21

ООО Планета Прінт

32