Государственное бюджетное профессиональное образовательное учреждение
«Байконурский электрорадиотехнический техникум имени М.И. Неделина»
(ГБ ПОУ «БЭРТТ»)

Методические рекомендации
по выполнению практических работ

по междисциплинарному курсу
«Технология разработки и защиты баз данных»

для специальности 09.02.03 «Программирование в компьютерных системах»

г. Байконур
2016 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Данные методические указания по выполнению практических работ составлены в соответствии с ФГОС по специальности СПО 09.02.03 «Программирование в компьютерных системах» (базовый уровень) по междисциплинарному курсу «Технология разработки и защиты баз данных».

С целью овладения указанным видом профессиональной деятельности и соответствующими профессиональными компетенциями обучающийся в ходе освоения междисциплинарного курса должен:

уметь:

- создавать объекты баз данных в современных системах управления базами данных и управлять доступом к этим объектам;
- работать с современными case-средствами проектирования баз данных;
- формировать и настраивать схему базы данных;
- разрабатывать прикладные программы с использованием языка SQL;
- создавать хранимые процедуры и триггеры на базах данных;
- применять стандартные методы для защиты объектов базы данных.

знать:

- основные положения теории баз данных, хранилищ данных, баз знаний;
- основные принципы построения концептуальной, логической и физической моделей данных;
- современные инструментальные средства разработки схемы базы данных;
- методы описания схем баз данных в современных системах управления базами данных (СУБД);
- структуры данных СУБД, общий подход к организации представлений, таблиц, индексов и кластеров;
- методы организации целостности данных;
- способы контроля доступа к данным и управления привилегиями;
- основные методы и средства защиты данных в базах данных;
- модели и структуры информационных систем;
- основы разработки приложений баз данных.

иметь практический опыт:

- работы с объектами базы данных в конкретной системе управления базами данных;

- построения концептуальной модели баз данных и разработки структуры баз данных;

- использования средств создания и заполнения базы данных;

- использования стандартных методов защиты объектов базы данных.

Перечень практических занятий

№	Тема
1	Построение ER-диаграммы
2	Построение схемы отношений
3	Использование базисных операций реляционной алгебры для выполнения запросов
4	Использование дополнительных операций реляционной алгебры для выполнения запросов
5	Построение сложных запросов с учетом свойств теоретико-множественных операций и правил формальной оптимизации
6	Обработка данных с помощью операции выборки SELECT
7	Редактирование базы данных с помощью SQL-инструкций UPDATE, INSERT, DELETE
8	Создание структуры базы данных с помощью SQL-инструкций

Правила выполнения практических работ

Студент перед выполнением практической работы должен строго выполнить весь объем домашней подготовки, указанный в описании соответствующей практической работы. Выполнению каждой работы может предшествовать проверка готовности студента, которая проводится преподавателем.

Отчет о практической работе должен содержать исчерпывающие систематизированные сведения о выполненной работе. Оформление отчета о практической работе должно быть предусмотрено заданием или планом выполнения работы. Отчет составляется исполнителем работы и защищается в индивидуальном порядке в сроки, установленные преподавателем.

Общими требованиями к отчету являются

- четкость и логическая последовательность изложения материала;
- убедительность аргументации;
- краткость и точность формулировок, исключающих возможность неоднозначного толкования;
- конкретность изложения результатов работы;
- обоснованность выводов.

Практическое занятие №1
Тема: «Построение ER-диаграммы»

Цель практического занятия: провести этап инфологического проектирования, построив ER-диаграмму предметной области.

Предварительная подготовка: лекционный материал по междисциплинарному курсу.

Количество часов: 2 часа

Постановка задачи

Построить ER-диаграмму в нотации Чена для предложенной предметной области.

Варианты заданий для построения ER-диаграммы:

1. Цветы. Разработать информационную систему, которая позволила заказать и отослать цветы по нужному адресу.
2. Банковский автомат. Разработать информационную систему, связанную с обслуживанием банковского автомата по выдаче денег по кредитной карточке.
3. Телефонная станция. Определить перечень услуг предоставляемой телефонной станцией с точки зрения пользователя телефона и с точки зрения обслуживающего персонала.
4. Почта. Определить перечень услуг предоставляемых почтовым отделением с точки зрения клиента, и сточки зрения работника почты.
5. Абитуриент. Разработать информационную систему, которая позволяла бы осуществлять ввод информации об абитуриенте, отслеживать сдачу им экзаменов и в результате выдавать необходимую информацию. Предусмотреть выдачу информации необходимой для абитуриента.
6. Отдел кадров. Разработать информационную систему, связанную с автоматизацией работы отдела кадров, предусмотреть наличие информации о трех видов сотрудников: студентов, преподавателей и учебно-вспомогательного персонала.
7. Деканат. Разработать информационную систему деканат для работы со студентами, предусмотреть выдачу необходимой информации для студентов и преподавателей.
8. Кафедра. Разработать информационную систему кафедра для работы со студентами, предусмотреть выдачу необходимой информации для студентов и преподавателей.
9. Магазин. Разработать информационную систему, связанную с обслуживанием покупателей в магазине, предусмотреть заказ товаров и обслуживание покупателей.
10. Больница. Разработать информационную систему, связанную с регистрацией больных, предусмотреть занесение информации о лечащем враче диагнозе, номере палаты, временем болезни и т.д.
11. Клуб собаководства. Разработать информационную систему по учету собаководов. Предусмотреть ведение личной карточки владельца и питомцев, деление на породы собак, возможность выбора и просмотра по различным критериям: порода, проживание владельца, возраст, награды и т.п.
12. Коллекция цветов. Разработать информационную систему для цветоводов-любителей с возможностью ведения каталога видов цветов (вид, название , шифр, описание, условия выращивания), адресов других коллекционеров и данных о их коллекциях.
13. Видео прокат. Разработать информационную систему для салона видео проката с возможностью ведения записей о кассетах (тип записи, когда снят, выпущен в прокат, стоимость проката, стоимость утери/покупки, время нахождения в салоне) и учета проката (список всех, бравших кассету, дата возврата, взнос, дата аренды).
14. Магазин подарков. Разработать информационную систему учета продаж/закупок/наличия товара в магазине. Предусмотреть возможность предварительного

заказа подарков, ведение статистики заказов/покупок/продаж, учесть возможность различных видов расчетов.

15. Оптовая торговля. Разработать информационную систему учета продаж/поступлений на склад/наличия товара на складе. Предусмотреть возможность предварительного заказа товара, ведение статистики заказов/покупок/продаж, ведение базы поставщиков/заказчиков.

16. Аптека. Разработать информационную систему учета продаж/закупок/наличия препаратов в аптеке. Предусмотреть возможность продажи препаратов по рецептам, ведение баланса покупок/продаж.

17. Продажа ЖД билетов. Разработать информационную систему кассового зала ЖД вокзала. Предусмотреть три вида обращений: бронирование, покупка в кассе, отказ. Вести статистику покупок/отказов по направлениям и общей загруженности направления.

18. Продажа авиа билетов. Разработать информационную систему кассового зала аэропорта. Предусмотреть три вида обращений: бронирование, покупка в кассе, отказ. Вести статистику покупок/отказов по направлениям и общей загруженности направления для различных авиакомпаний.

19. Детский клуб (кружки, секции …). Разработать информационную систему статистического анализа работы детского клуба. Предусмотреть возможность добавления/удаления/переименования кружков, ведение расписания занятий и посещаемости кружка; получение информации о составе занимающихся групп, преподавателях и т.д.

20. Библиотека. Разработать информационную систему для ведения каталога книг/читателей, поисковой системы, системы предварительных заказов на приобретение книг, а так же системы предварительной записи на использование дефицитной литературы и просмотра очереди.

21. Рекламное агентство. Разработать информационную систему для ведения бухгалтерии рекламного агентства. Предусмотреть учет заказчиков, посредников, исполнителей.

22. Объявления в газете. Разработать информационную систему для создания рекламных страниц газеты. Предусмотреть учет рекламодателей (юридических и физических лиц), различные разделы рекламы, систему скидок на цену объявлений по различным категориям.

23. Продажа недвижимости. Разработать информационную систему для продажи/покупки недвижимости, ведение базы продавцов и покупателей с регистрацией покупки/продажи, регистрацией номера свидетельства о праве собственности. Предусмотреть возможность выдачи статистики о изменении стоимости квадратного метра по районам города за заданный период времени.

24. Валютные операции. Разработать информационную систему для продажи/покупки различных валют, ведение базы продавцов и покупателей валют с регистрацией номера счета-квитанции о покупке (продаже). Предусмотреть возможность выдачи статистики о изменении курса валют за заданный период времени.

25. Домашняя бухгалтерия. Разработать информационную систему для ведения домашней бухгалтерии одной семьи. Предусмотреть различные типы доходов и расходов, ведение баланса, заполнение налоговых документов, счетов.

26. Регистрация нарушений в ГАИ. Разработать информационную систему для регистрации нарушений, предусмотреть ведение записей о виде нарушения, схеме ДТП, номере машины, ее модели, года выпуска, владельце.

27. Регистрация машин в ГАИ. Разработать информационную систему для выдачи информации о государственном регистрационном знаке, марка, модель, тип ТС, его идентификационном номере машины, года выпуска, номере двигателя, номере кузова владельце, предыдущем владельце, номере техпаспорта, серии и номере свидетельства о регистрации ТС и дате регистрации.

28. Система тестирования в ГАИ. Разработать информационную систему для сдачи экзамена в ГАИ. Предусмотреть ведение записей о сдавших и о не сдавших экзамен, выдачу экзаменационных тестов в случайном порядке и т.д.

29. Коммунальные платежи. Разработать информационную систему для расчета и выдачи квитанций об оплате коммунальных платежей, регистрации оплаты.

30. Телефонный справочник. Разработать информационную систему для регистрации, поиска и выдачи информации об абонентах телефонной сети города.

Методика выполнения практической работы

Модель «сущность-связь» была предложена в 1976 году Питером Пин-Шен Ченом (англ. Peter Pin-Shen Chen), американским профессором компьютерных наук в университете штата Луизиана.

Модель сущность-связь (ER-модель) (англ. entity-relationship model, ERM) — модель данных, позволяющая описывать концептуальные схемы предметной области.

ER-модель используется при высокоуровневом (концептуальном) проектировании баз данных. С её помощью можно выделить ключевые сущности и обозначить связи, которые могут устанавливаться между этими сущностями.

Во время проектирования баз данных происходит преобразование ER-модели в конкретную схему базы данных на основе выбранной модели данных (реляционной, объектной, сетевой или др.).

ER-модель представляет собой формальную конструкцию, которая сама по себе не предписывает никаких графических средств её визуализации. В качестве стандартной графической нотации, с помощью которой можно визуализировать ER-модель, была предложена диаграмма сущность-связь (ER-диаграмма) (англ. entity-relationship diagram, ERD).

Нотация Питера Чена

Множества сущностей изображаются в виде прямоугольников, множества отношений изображаются в виде ромбов. Если сущность участвует в отношении, они связаны линией. Если отношение не является обязательным, то линия пунктирная. Атрибуты изображаются в виде овалов и связываются линией с одним отношением или с одной сущностью.

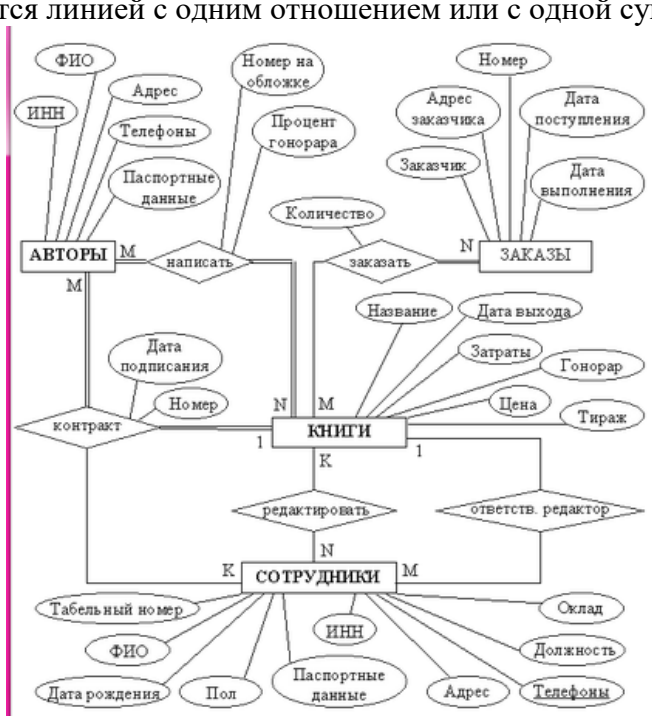


Рисунок 1. Диаграмма «сущность-связь» в нотации Чена

Нотация Crow's Foot

Данная нотация была предложена Гордоном Эверестом(англ. Gordon Everest) под названием Inverted Arrow («перевёрнутая стрелка»), однако сейчас чаще называемая Crow's Foot («воронья лапка») или Fork («вилка»).

Согласно данной нотации, сущность изображается в виде прямоугольника, содержащем её имя, выражаемое существительным. Имя сущности должно быть уникальным в рамках одной модели. При этом имя сущности - это имя типа, а не конкретного экземпляра данного типа. Экземпляром сущности называется конкретный представитель данной сущности.

Связь изображается линией, которая связывает две сущности, участвующие в отношении. Степень конца связи указывается графически, множественность связи изображается в виде «вилки» на конце связи. Модальность связи также изображается графически - необязательность связи помечается кружком на конце связи. Именование обычно выражается одним глаголом в изъявительном наклонении настоящего времени: «Имеет», «Принадлежит» и т. д.; или глаголом с поясняющими словами: «Включает в себя», и т.п. Наименование может быть одно для всей связи или два для каждого из концов связи. Во втором случае, название левого конца связи указывается над линией связи, а правого – под линией. Каждое из названий располагаются рядом с сущностью, к которой оно относится.

Атрибуты сущности записываются внутри прямоугольника, изображающего сущность и выражаются существительным в единственном числе (возможно, с уточняющими словами). Среди атрибутов выделяется ключ сущности — неизбыточный набор атрибутов, значения которых в совокупности являются уникальными для каждого экземпляра сущности.

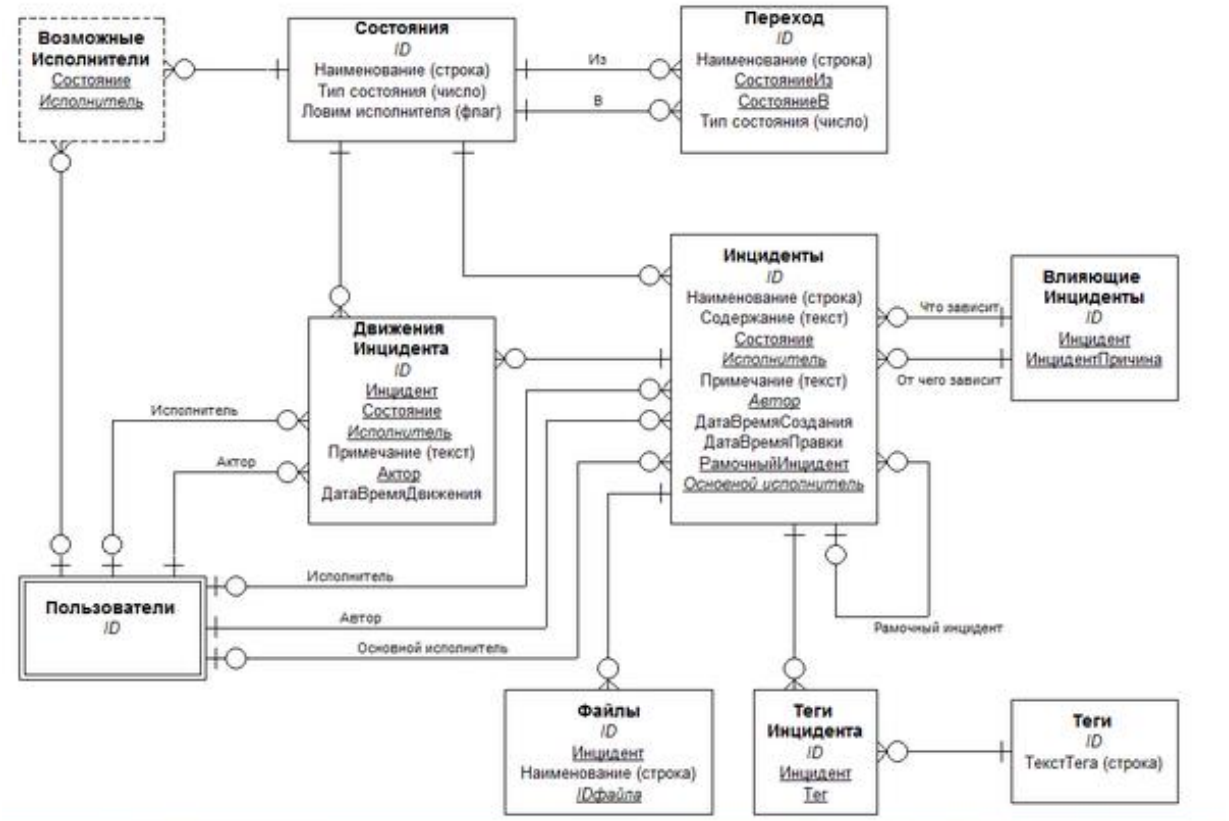


Рисунок 2 - Диаграмма «сущность-связь» в нотации Crow's Foot

Инструментальные средства для построения ER-диаграмм.

На сегодняшний день Российский рынок программного обеспечения располагает следующими наиболее развитыми CASE-средствами: Vantage Team Builder (Westmount I-CASE); Designer/2000; Silverrun; ERwin; S-Designor; CASE.Аналитик.

Отчет должен содержать:
1. Название, цель и задание практической работы;
2. ER-диаграмма предметной области;
3. Ответы на контрольные вопросы.

Контрольные вопросы:
1. Что такое «инфологическая модель», «сущность», «атрибут», «связь»?
2. Для чего проводится этап инфологического проектирования предметной области?
3. Что такое «нотация»?

Практическое занятие № 2
Тема: «Построение схемы отношений»

Цель практического занятия: провести этап концептуального проектирования, выполнив нормализацию отношений предметной области.

Предварительная подготовка: лекционный материал по междисциплинарному курсу.

Количество часов: 2 часа

Постановка задачи

Построить схему данных на основе нормализованных (до 3НФ) отношений для предложенной (по варианту) предметной области.

Варианты заданий для построения схемы данных

1. Цветы. Разработать информационную систему, которая позволила заказать и отослать цветы по нужному адресу.
2. Банковский автомат. Разработать информационную систему, связанную с обслуживанием банковского автомата по выдаче денег по кредитной карточке.
3. Телефонная станция. Определить перечень услуг предоставляемой телефонной станцией с точки зрения пользователя телефона и с точки зрения обслуживающего персонала.
4. Почта. Определить перечень услуг предоставляемых почтовым отделением с точки зрения клиента, и сточки зрения работника почты.
5. Абитуриент. Разработать информационную систему, которая позволяла бы осуществлять ввод информации об абитуриенте, отслеживать сдачу им экзаменов и в результате выдавать необходимую информацию. Предусмотреть выдачу информации необходимой для абитуриента.
6. Отдел кадров. Разработать информационную систему, связанную с автоматизацией работы отдела кадров, предусмотреть наличие информации о трех видов сотрудников: студентов, преподавателей и учебно-вспомогательного персонала.
7. Деканат. Разработать информационную систему деканат для работы со студентами, предусмотреть выдачу необходимой информации для студентов и преподавателей.
8. Кафедра. Разработать информационную систему кафедра для работы со студентами, предусмотреть выдачу необходимой информации для студентов и преподавателей.
9. Магазин. Разработать информационную систему, связанную с обслуживанием покупателей в магазине, предусмотреть заказ товаров и обслуживание покупателей.
10. Больница. Разработать информационную систему, связанную с регистрацией больных, предусмотреть занесение информации о лечащем враче диагнозе, номере палаты, временем болезни и т.д.
11. Клуб собаководства. Разработать информационную систему по учету собаководов. Предусмотреть ведение личной карточки владельца и питомцев, деление на породы собак, возможность выбора и просмотра по различным критериям: порода, проживание владельца, возраст, награды и т.п.
12. Коллекция цветов. Разработать информационную систему для цветоводов-любителей с возможностью ведения каталога видов цветов (вид, название , шифр, описание, условия выращивания), адресов других коллекционеров и данных о их коллекциях.
13. Видео прокат. Разработать информационную систему для салона видео проката с возможностью ведения записей о кассетах (тип записи, когда снят, выпущен в прокат, стоимость проката, стоимость утери/покупки, время нахождения в салоне) и учета проката (список всех, бравших кассету, дата возврата, взнос, дата аренды).

14. Магазин подарков. Разработать информационную систему учета продаж/закупок/наличия товара в магазине. Предусмотреть возможность предварительного заказа подарков, ведение статистики заказов/покупок/продаж, учесть возможность различных видов расчетов.

15. Оптовая торговля. Разработать информационную систему учета продаж/поступлений на склад/наличия товара на складе. Предусмотреть возможность предварительного заказа товара, ведение статистики заказов/покупок/продаж, ведение базы поставщиков/заказчиков.

16. Аптека. Разработать информационную систему учета продаж/закупок/наличия препаратов в аптеке. Предусмотреть возможность продажи препаратов по рецептам, ведение баланса покупок/продаж.

17. Продажа ЖД билетов. Разработать информационную систему кассового зала ЖД вокзала. Предусмотреть три вида обращений: бронирование, покупка в кассе, отказ. Вести статистику покупок/отказов по направлениям и общей загруженности направления.

18. Продажа авиа билетов. Разработать информационную систему кассового зала аэропорта. Предусмотреть три вида обращений: бронирование, покупка в кассе, отказ. Вести статистику покупок/отказов по направлениям и общей загруженности направления для различных авиакомпаний.

19. Детский клуб (кружки, секции …). Разработать информационную систему статистического анализа работы детского клуба. Предусмотреть возможность добавления/удаления/переименования кружков, ведение расписания занятий и посещаемости кружка; получение информации о составе занимающихся групп, преподавателях и т.д.

20. Библиотека. Разработать информационную систему для ведения каталога книг/читателей, поисковой системы, системы предварительных заказов на приобретение книг, а так же системы предварительной записи на использование дефицитной литературы и просмотра очереди.

21. Рекламное агентство. Разработать информационную систему для ведения бухгалтерии рекламного агентства. Предусмотреть учет заказчиков, посредников, исполнителей.

22. Объявления в газете. Разработать информационную систему для создания рекламных страниц газеты. Предусмотреть учет рекламодателей (юридических и физических лиц), различные разделы рекламы, систему скидок на цену объявлений по различным категориям.

23. Продажа недвижимости. Разработать информационную систему для продажи/покупки недвижимости, ведение базы продавцов и покупателей с регистрацией покупки/продажи, регистрацией номера свидетельства о праве собственности. Предусмотреть возможность выдачи статистики о изменении стоимости квадратного метра по районам города за заданный период времени.

24. Валютные операции. Разработать информационную систему для продажи/покупки различных валют, ведение базы продавцов и покупателей валют с регистрацией номера счета-квитанции о покупке (продаже). Предусмотреть возможность выдачи статистики о изменении курса валют за заданный период времени.

25. Домашняя бухгалтерия. Разработать информационную систему для ведения домашней бухгалтерии одной семьи. Предусмотреть различные типы доходов и расходов, ведение баланса, заполнение налоговых документов, счетов.

26. Регистрация нарушений в ГАИ. Разработать информационную систему для регистрации нарушений, предусмотреть ведение записей о виде нарушения, схеме ДТП, номере машины, ее модели, года выпуска, владельце.

27. Регистрация машин в ГАИ. Разработать информационную систему для выдачи информации о государственном регистрационном знаке, марка, модель, тип ТС, его идентификационном номере машины, года выпуска, номере двигателя, номере кузова

владельце, предыдущем владельце, номере техпаспорта, серии и номере свидетельства о регистрации ТС и дате регистрации.

28. Система тестирования в ГАИ. Разработать информационную систему для сдачи экзамена в ГАИ. Предусмотреть ведение записей о сдавших и о не сдавших экзамен, выдачу экзаменационных тестов в случайном порядке и т.д.

Методика выполнения практической работы

В таблицах реляционных БД возможно дублирование информации. Чтобы этого избежать проводят процесс нормализации исходных таблиц.

Нормализация – это разбиение таблицы на две или более, обладающих лучшими свойствами при включении, изменении и удалении данных.

Каждая таблица в реляционной БД удовлетворяет условию, в соответствии с которым в позиции на пересечении каждой строки и столбца таблицы всегда находится единственное атомарное значение, и никогда не может быть множества таких значений. Любая таблица, удовлетворяющая этому условию, называется нормализованной. Фактически, ненормализованные таблицы, т.е. таблицы, содержащие повторяющиеся группы, даже не допускаются в реляционной БД.

В результате нормализации необходимо, чтобы столбцы каждой таблицы зависели от первичного ключа и не зависели друг от друга.

Заключительным этапом нормализации является построение концептуальной схемы данных.

Пример. Концептуальная модель БД «Дневник успеваемости учеников»

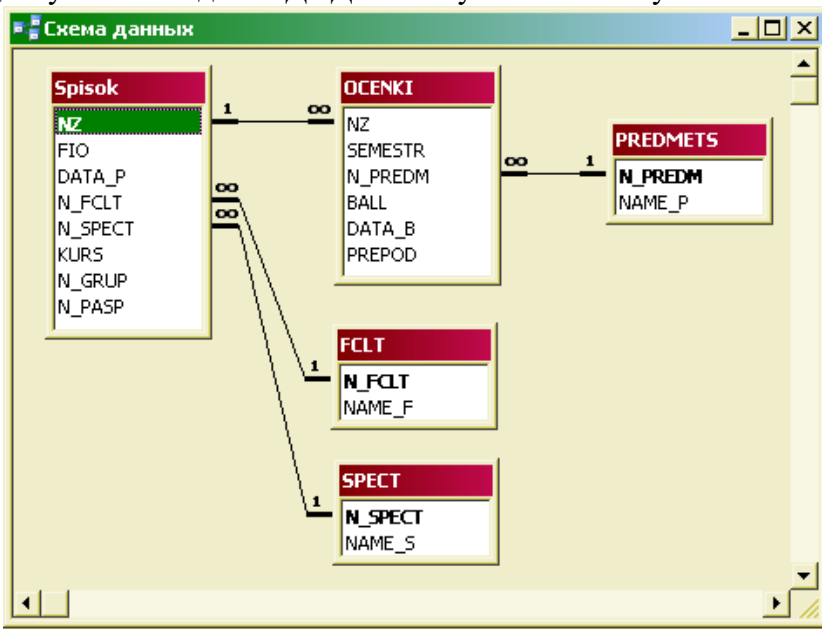


Рисунок 1. Схема БД

Различают 5 НФ: 1НФ, 2НФ, 3НФ, Бойса-Кодда (НФБК), 5НФ.

Таблица находится в первой нормальной форме (1НФ) тогда и только тогда, когда ни одна из ее строк не содержит в любом своем поле более одного значения и ни одно из ее ключевых полей не пусто.

Таблица находится во второй нормальной форме (2НФ), если она удовлетворяет определению 1НФ и все ее поля, не входящие в первичный ключ, связаны полной функциональной зависимостью с первичным ключом.

Таблица находится в третьей нормальной форме (3НФ), если она удовлетворяет определению 2НФ и не одно из ее неключевых полей не зависит функционально от любого другого неключевого поля.

Таблица находится в нормальной форме Бойса-Кодда (НФБК), если и только если любая функциональная зависимость между его полями сводится к полной функциональной зависимости от возможного ключа.

Полной декомпозицией таблицы называют такую совокупность произвольного числа ее проекций, соединение которых полностью совпадает с содержимым таблицы.

Таблица находится в пятой нормальной форме (5НФ) тогда и только тогда, когда в каждой ее полной декомпозиции все проекции содержат возможный ключ. Таблица, не имеющая ни одной полной декомпозиции, также находится в 5НФ.

Четвертая нормальная форма (4НФ) является частным случаем 5НФ, когда полная декомпозиция должна быть соединением ровно двух проекций. Весьма не просто подобрать реальную таблицу, которая находилась бы в 4НФ, но не была бы в 5НФ.

Отчет должен содержать:

1. Название, цель и задание практической работы;
2. Схема-данных предметной области;
3. Ответы на контрольные вопросы.

Контрольные вопросы

1. Что такое «концептуальная модель», «отношение», «первичный ключ», «внешний ключ», «связь», «поле», «запись»?
2. Для чего проводится нормализация отношений предметной области?
3. Какие разновидности нормальных форм существуют?

Практическое занятие № 3
Тема: «Использование базисных операций реляционной алгебры для выполнения запросов»

Цель практического занятия: изучить принципы работы теоретико-множественных операторов, научиться использовать операторы реляционной алгебры при выполнении операций над отношениями.

Предварительная подготовка: лекционный материал по междисциплинарному курсу.

Количество часов: 2 часа

Постановка задачи

Задание. Даны отношения A, B и C. Выполнить над ними операции, в соответствии с вариантом задания:

I Вариант - объединение отношений A и B, пересечение отношений A и B, вычитание отношений (A-B), декартово произведение отношений A и C.

II Вариант - объединение отношений A и B, пересечение отношений A и B, вычитание отношений (B-A)), декартово произведение отношений B и C.

Отношение A

Id_номер	Наименование	Производитель	Цена, руб.	Количество, шт.
2913	Стол компьютерный	АО «Офис Мода»	2 950	5
2914	Тумба под ТВ	ООО «Строй Сервис»	1 900	8
2915	Кухонный уголок	ООО «Уют»	5 400	4
2916	Стул офисный	ТОО «Дизайн»	2 200	7

Отношение B

Инв. номер	Наименование	Фирма	Цена, руб.	Количество, шт.
2911	Шкаф - купе	АО «Офис Мода»	5 800	2
2914	Тумба под ТВ	ООО «Строй Сервис»	1 900	8
2915	Кухонный уголок	ООО «Уют»	5 400	4
2917	Стул офисный	ТОО «Дизайн»	1 650	3

Отношение C

Номер п/п	Склад
1	Головной
2	ЦУП
3	ЧП

Методика выполнения практической работы

1 Реляционная алгебра

Реляционная алгебра представляет собой набор операторов, использующих отношения в качестве аргументов, и возвращающие отношения в качестве результата. Таким образом, реляционный оператор f выглядит как функция с отношениями в качестве аргументов:

$$R = f(R_1, R_2, \ldots, R_n)$$

Реляционная алгебра является замкнутой, т.к. в качестве аргументов в реляционные операторы можно подставлять другие реляционные операторы, подходящие по типу:

$$R = f(f_1(R_{11}, R_{12}, \ldots), f_2(R_{21}, R_{22}, \ldots), \ldots)$$

Таким образом, в реляционных выражениях можно использовать вложенные выражения сколь угодно сложной структуры.

Каждое отношение обязано иметь уникальное имя в пределах базы данных. Имя отношения, полученного в результате выполнения реляционной операции, определяется в левой части равенства. Однако можно не требовать наличия имен от отношений, полученных в результате реляционных выражений, если эти отношения подставляются в качестве аргументов в другие реляционные выражения. Такие отношения будем называть неименованными отношениями. Неименованные отношения реально не существуют в базе данных, а только вычисляются в момент вычисления значения реляционного оператора.

Традиционно, вслед за Коддом, определяют восемь реляционных операторов, объединенных в две группы.

Теоретико-множественные операторы:

Объединение

Пересечение

Вычитание

Декартово произведение

Специальные реляционные операторы:

Выборка

Проекция

Соединение

Деление

Не все они являются независимыми, т.е. некоторые из этих операторов могут быть выражены через другие реляционные операторы.

Отношения, совместимые по типу

Определение 1. Будем называть отношения совместимыми по типу, если они имеют идентичные заголовки, а именно,

- Отношения имеют одно и то же множество имен атрибутов, т.е. для любого атрибута в одном отношении найдется атрибут с таким же наименованием в другом отношении,
- Атрибуты с одинаковыми именами определены на одних и тех же доменах.

Некоторые отношения не являются совместимыми по типу, но становятся таковыми после некоторого переименования атрибутов. Для того чтобы такие отношения можно было использовать в реляционных операторах, вводится вспомогательный оператор переименования атрибутов.

Оператор переименования атрибутов

Оператор переименования атрибутов имеет следующий синтаксис:

R RENAME $Atr_1, Atr_2, \ldots$ AS $NewAtr_1, NewAtr_2, \ldots$

где

R - отношение,

$Atr_1, Atr_2, \ldots$ - исходные имена атрибутов,

$NewAtr_1, NewAtr_2, \ldots$ - новые имена атрибутов.

В результате применения оператора переименования атрибутов получаем новое отношение, с измененными именами атрибутов.

Пример 1.

Следующий оператор возвращает неименованное отношение, в котором атрибут City_Num переименован в CityId:

City RENAME *City_Num* AS *CityId*

2 Теоретико-множественные операторы

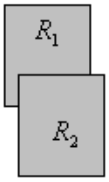
1) Объединение

Отношения-операнды в этом случае должны быть определены по одной схеме. Результирующее отношение содержит все строки операндов за исключением повторяющихся.

Объединение / UNION /

Обозначение	Определение	LEAP
$R_1 \cup R_2$	$\{r : r \in R_1 \vee r \in R_2\}$	r = (R1) union (R2)

Пример:



$$R[Q,T] \cup S = \begin{bmatrix} 5 & a \\ 3 & a \\ 9 & a \\ 1 & b \\ 2 & b \\ 4 & b \end{bmatrix} \cup \begin{bmatrix} 5 & a \\ 10 & b \\ 15 & c \\ 2 & d \\ 6 & a \\ 1 & b \end{bmatrix} = \begin{bmatrix} 5 & a \\ 3 & a \\ 9 & a \\ 1 & b \\ 2 & b \\ 4 & b \\ 10 & b \\ 15 & c \\ 2 & d \\ 6 & a \end{bmatrix}$$

Определение 2. Объединением двух совместимых по типу отношений A и B называется отношение с тем же заголовком, что и у отношений A и B, и телом, состоящим из кортежей, принадлежащих или A, или B, или обоим отношениям.

Синтаксис операции объединения:

A UNION B

Замечание. Объединение, как и любое отношение, не может содержать одинаковых кортежей. Поэтому, если некоторый кортеж входит и в отношение A, и отношение B, то в объединение он входит один раз.

Пример 2. Пусть даны два отношения A и B с информацией о сотрудниках:

Табельный номер	Фамилия	Зарплата
1	Иванов	1000
2	Петров	2000
3	Сидоров	3000

Таблица 1 Отношение A

Табельный номер	Фамилия	Зарплата
1	Иванов	1000
2	Пушников	2500
4	Сидоров	3000

Таблица 2 Отношение B

Объединение отношений A и B будет иметь вид:

Табельный номер	Фамилия	Зарплата

1	Иванов	1000
2	Петров	2000
3	Сидоров	3000
2	Пушников	2500
4	Сидоров	3000

Таблица 3 Отношение A UNION B

Замечание. Как видно из приведенного примера, потенциальные ключи, которые были в отношениях A и B не наследуются объединением этих отношений. Поэтому, в объединении отношений A и B атрибут "Табельный номер" может содержать дубликаты значений. Если бы это было не так, и ключи наследовались бы, то это противоречило бы понятию объединения как "объединение множеств". Конечно, объединение отношений A и B имеет, как и любое отношение, потенциальный ключ, например, состоящий из всех атрибутов.

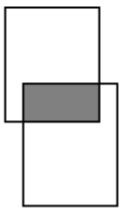
2) Пересечение

На входе операции два отношения, определенные по одной схеме. На выходе - отношение, содержащие кортежи, которые присутствуют в обоих исходных отношениях.

Пересечение / INTERSECT /

Обозначение	Определение	LEAP
$R_1 \cap R_2$	$\{r : r \in R_1 \wedge r \in R_2\}$	r = (R1) intersect (R2)

Пример:



$$R[Q,T] \cap S = \begin{bmatrix} 5 & a \\ 3 & a \\ 9 & a \\ 1 & b \\ 2 & b \\ 4 & b \end{bmatrix} \cap \begin{bmatrix} 5 & a \\ 10 & b \\ 15 & c \\ 2 & d \\ 6 & a \\ 1 & b \end{bmatrix} = \begin{bmatrix} 5 & a \\ 1 & b \end{bmatrix}$$

Определение 3. Пересечением двух совместимых по типу отношений A и B называется отношение с тем же заголовком, что и у отношений A и B, и телом, состоящим из кортежей, принадлежащих одновременно обоим отношениям A и B.

Синтаксис операции пересечения:

A INTERSECT B

Пример 3. Для тех же отношений A и B, что и в предыдущем примере пересечение имеет вид:

Табельный номер	Фамилия	Зарплата
1	Иванов	1000

Таблица 4 Отношение A INTERSECT B

Замечание. Казалось бы, что в отличие от операции объединения, потенциальные ключи могли бы наследоваться пересечением отношений. Однако это не так. Вообще, никакие реляционные операторы не передают результатирующему отношению никаких

данных о потенциальных ключах. В качестве причины этого можно было бы привести тривиальное соображение, что так получается более просто и симметрично - все операторы устроены одинаково. На самом деле причина более глубока, и заключается в том, что потенциальный ключ - семантическое понятие, отражающее различимость объектов предметной области. Наличие потенциальных ключей не выводится из структуры отношения, а явно задается для каждого отношения, исходя из его смысла. Реляционные же операторы являются формальными операциями над отношениями и выполняются одинаково, независимо от смысла данных, содержащихся в отношениях. Поэтому, реляционные операторы ничего не могут "знать" о смысле данных. Трактовка результата реляционных операций - дело пользователя.

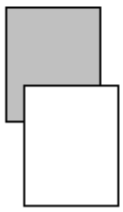
3) Вычитание

Операция во многом похожая на ПЕРЕСЕЧЕНИЕ, за исключением того, что в результирующем отношении содержатся кортежи, присутствующие в первом и отсутствующие во втором исходных отношениях.

Разность / SET DIFFERENCE /

Обозначение	Определение	LEAP
$R_1 - R_2$	$\{r : r \in R_1 \wedge r \notin R_2\}$	r = (R1) difference (R2) r = (R1) minus (R2)

Пример:



$$R[Q,T] - S = \begin{bmatrix} 5 & a \\ 3 & a \\ 9 & a \\ 1 & b \\ 2 & b \\ 4 & b \end{bmatrix} \cap \begin{bmatrix} 5 & a \\ 10 & b \\ 15 & c \\ 2 & d \\ 6 & a \\ 1 & b \end{bmatrix} = \begin{bmatrix} 3 & a \\ 9 & a \\ 2 & b \\ 4 & b \end{bmatrix}$$

Определение 4. Вычитанием двух совместимых по типу отношений A и B называется отношение с тем же заголовком, что и у отношений A и B, и телом, состоящим из кортежей, принадлежащих отношению A и не принадлежащих отношению B.

Синтаксис операции вычитания:

A MINUS B

Пример 4. Для тех же отношений A и B, что и в предыдущем примере вычитание имеет вид:

Табельный номер	Фамилия	Зарплата
2	Петров	2000
3	Сидоров	3000

Таблица 5 Отношение A MINUS B

4) Декартово произведение

Входные отношения могут быть определены по разным схемам. Схема результирующего отношения включает все атрибуты исходных. Кроме того:

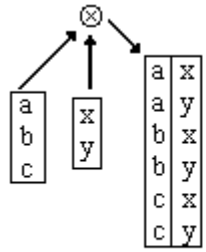
- степень результирующего отношения равна сумме степеней исходных отношений

○ мощность результирующего отношения равна произведению мощностей исходных отношений.

Декартово произведение / CARTESIAN PRODUCT /

Обозначение	Определение	LEAP
$R_1 \otimes R_2$	$\{(r_1 \| r_2) : r_1 \in R_1 \wedge r_2 \in R_2\}$	r = (R1) product (R2)

Пример:



$$R[M,T] \otimes \left(R[Q,T] \cap S\right) = \begin{bmatrix} x & a \\ y & a \\ z & a \\ w & b \end{bmatrix} \otimes \begin{bmatrix} 5 & a \\ 1 & b \end{bmatrix} = \begin{bmatrix} x & a & 5 & a \\ x & a & 1 & b \\ y & a & 5 & a \\ y & a & 1 & b \\ z & a & 5 & a \\ z & a & 1 & b \\ w & b & 5 & a \\ w & b & 1 & b \end{bmatrix}$$

Определение 5. Декартовым произведением двух отношений $A(A_1, A_2, \ldots, A_n)$ и $B(B_1, B_2, \ldots, B_m)$ называется отношение, заголовок которого является сцеплением заголовков отношений A и B:

$(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m)$,

а тело состоит из кортежей, являющихся сцеплением кортежей отношений A и B:

$(a_1, a_2, \ldots a_n, b_1, b_2, \ldots b_m)$,

таких, что $(a_1, a_2, \ldots, a_n) \in A$, $(b_1, b_2, \ldots b_m) \in B$.

Синтаксис операции декартового произведения:

A TIMES B

Замечание. Мощность произведения A TIMES B равна произведению мощностей отношений A и B, т.к. каждый кортеж отношения A соединяется с каждым кортежем отношения B.

Замечание. Если в отношения A и B имеются атрибуты с одинаковыми наименованиями, то перед выполнением операции декартового произведения такие атрибуты необходимо переименовать.

Замечание. Перемножать можно любые два отношения, совместимость по типу при этом не требуется.

Пример 5. Пусть даны два отношения A и B с информацией о поставщиках и деталях:

Номер поставщика	Наименование поставщика
1	Иванов
2	Петров
3	Сидоров

Таблица 6 Отношение A (Поставщики)

Номер детали	Наименование детали
1	Болт
2	Гайка
3	Винт

Таблица 7 Отношение B (Детали)

Декартово произведение отношений A и B будет иметь вид:

Номер поставщика	Наименование поставщика	Номер детали	Наименование детали
1	Иванов	1	Болт
1	Иванов	2	Гайка
1	Иванов	3	Винт
2	Петров	1	Болт
2	Петров	2	Гайка
2	Петров	3	Винт
3	Сидоров	1	Болт
3	Сидоров	2	Гайка
3	Сидоров	3	Винт

Таблица 8 Отношение A TIMES B

Замечание. Сама по себе операция декартового произведения не очень важна, т.к. она не дает никакой новой информации, по сравнению с исходными отношениями. Для реальных запросов эта операция почти никогда не используется. Однако операция декартового произведения важна для выполнения специальных реляционных операций, о которых речь пойдет ниже.

Отчет должен содержать:
1. Название, цель и задание практической работы;
2. Отношения, полученные в результате выполнения базовых реляционных операторов;
3. Ответы на контрольные вопросы.

Контрольные вопросы
1. Что изучает реляционная алгебра?
2. Какие существуют группы реляционных операторов?
3. Какие реляционные операторы относятся к базовой группе?

Практическое занятие № 4

Тема: «Использование дополнительных операций реляционной алгебры для выполнения запросов»

Цель практического занятия: изучить принципы работы теоретико-множественных операторов, научиться использовать операторы реляционной алгебры при выполнении операций над отношениями.

Предварительная подготовка: лекционный материал по междисциплинарному курсу.

Количество часов: 2 часа

Постановка задачи

Даны отношения А и В. Выполнить над ними операции реляционной алгебры в соответствии с заданием, указанным в таблице 16.

Таблица 1.Варианты заданий

№ задания / № варианта	1	2	3
Вариант 1	А WHERE (Часовая тарифная ставка >= 245) OR (Класс = высший)	А[Класс, Категория]	А[ФИО водителя = ФИО бригадира]Б
Вариант 2	Б WHERE (Наименование ГСМ = АИ-92) AND (ФИО бригадира = Ершиков М.Т.)	Б[ID ТС, Гос. номер, Код марки ТС]	А[ФИО водителя ≠ ФИО бригадира]Б

Отношение А. Картотека водителя

ID водителя	Табельный номер	ФИО водителя	Класс	Номер удостоверения	Категория	Часовая тарифная ставка
1	000456	Агафонов П.И.	Высший	1562890	С	250
2	006789	Волков К.Н.	Средний	2340096	А	245
3	110934	Дудкин Е.Л.	Высший	1568899	Б	190
4	211456	Ершиков М.Т.	Высший	5410093	С	176

Отношение Б. Транспортное средство

ID ТС	Гос. номер	Код марки ТС	ФИО бригадира	Наименование ГСМ
1	Р234ЛД	М3317	Агафонов П.И.	АИ-92
2	Р234ЛД	М3317	Ершиков М.Т.	АИ-92
3	РТ140И	М0092	Зенкин Р.Р.	А-56
4	РТ140И	М0092	Ли Э.Ж.	А-56

Методика выполнения практической работы

Специальные реляционные операторы

К специальным реляционным операторам относят операторы:

Выборки

Проекции

Соединения

Деления

Не все они являются независимыми, т.е. некоторые из этих операторов могут быть выражены через другие реляционные операторы.

1 Выборка (ограничение, селекция)

На входе используется одно отношение, результат - новое отношение, построенное по той же схеме, содержащее подмножество кортежей исходного отношения, удовлетворяющих условию выборки.

Выборка / SELECT /

Обозначение	Определение	LEAP
$R[A\ \theta v]$ $R[A_1\ \theta A_2]$	$\{r : r \in R \wedge (r[A]\ \theta v)\}$ $\{r : r \in R \wedge (r[A_1]\ \theta r[A_2])\}$	r = select (R) ((cond) bool (cond))

Пример:

$$P[D_2 = 11] = \begin{bmatrix} 1 & 11 & x \\ 2 & 11 & y \\ 3 & 11 & z \end{bmatrix}$$

Определение 1. Выборкой (ограничением, селекцией) на отношении A с условием c называется отношение с тем же заголовком, что и у отношения A, и телом, состоящем из кортежей, значения атрибутов которых при подстановке в условие c дают значение ИСТИНА. c представляет собой логическое выражение, в которое могут входить атрибуты отношения A и (или) скалярные выражения.

В простейшем случае условие c имеет вид $X \Theta Y$, где Θ - один из операторов сравнения ($=, \neq, <, \leq, >, \geq$ и т.д.), а X и Y - атрибуты отношения A или скалярные значения. Такие выборки называются Θ -выборки (тэта-выборки) или Θ -ограничения, Θ -селекции.

Синтаксис операции выборки:

A WHERE c ,

или

A WHERE $X \Theta Y$

Пример 1. Пусть дано отношение A с информацией о сотрудниках:

Табельный номер	Фамилия	Зарплата
1	Иванов	1000
2	Петров	2000
3	Сидоров	3000

Таблица 1 Отношение А

Результат выборки A WHERE Зарплата < 3000 будет иметь вид:

Табельный	Фа	За

номер	милия	рплата
1	Ив анов	10 00
2	Пе тров	20 00

Таблица 2 Отношение A WHERE Зарплата<3000

Смысл операции выборки очевиден - выбрать кортежи отношения, удовлетворяющие некоторому условию. Таким образом, операция выборки дает "горизонтальный срез" отношения по некоторому условию.

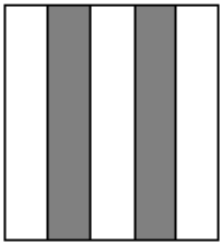
2 Проекция

Операция проекции представляет собой выборку из каждого кортежа отношения значений атрибутов, входящих в список A, и удаление из полученного отношения повторяющихся строк.

Проекция / PROJECT /

Обозначение	Определение	LEAP
$R[A]$	$\{r[A] : r \in R\}$	r = project (R) $(A_1, A_2, \ldots, A_n)$

Пример:



$$R[M,T] = \begin{bmatrix} x & a \\ y & a \\ z & a \\ w & b \\ w & b \\ w & b \end{bmatrix} = \begin{bmatrix} x & a \\ y & a \\ z & a \\ w & b \end{bmatrix}$$

Определение 2. Проекцией отношения A по атрибутам X, Y, …, Z, где каждый из атрибутов принадлежит отношению A, называется отношение с заголовком (X, Y, …, Z) и телом, содержащим множество кортежей вида (X, Y, …, Z), таких, для которых в отношении A найдутся кортежи со значением атрибута X равным x, значением атрибута Y равным y, …, значением атрибута Z равным z.

Синтаксис операции проекции:

$A[X, Y, \ldots, Z]$

Замечание. Операция проекции дает "вертикальный срез" отношения, в котором удалены все возникшие при таком срезе дубликаты кортежей.

Пример 2. Пусть дано отношение A с информацией о поставщиках, включающих наименование и месторасположение:

Номер поставщика	Наименование поставщика	Город поставщика
1	Иванов	Уфа
2	Петров	Москва
3	Сидоров	Москва
4	Сидоров	Челябинск

Таблица 3 Отношение A (Поставщики)

Проекция A[Город поставщика] будет иметь вид:

Город поставщика
Уфа
Москва
Челябинск

Таблица 4 Отношение A[Город поставщика]

3 Соединение

Данная операция имеет сходство с ДЕКАРТОВЫМ ПРОИЗВЕДЕНИЕМ. Однако, здесь добавлено условие, согласно которому вместо полного произведения всех строк в результирующее отношение включаются только строки, удовлетворяющие определенному соотношению между атрибутами соединения (A1,A2) соответствующих отношений.

Соединение / JOIN /

Обозначение	Определение
$R_1 \left[A_1 \ \theta A_2\right] R_2$	$\left\{(r_1 \| r_2) : r_1 \in R_1 \wedge r_2 \in R_2 \wedge \left(r_1\left[A_1\right] \theta r_2\left[A_2\right]\right)\right\}$

LEAP : r = join (R1) (R2) ((cond) bool (cond))

Пример:

$$P\left[D_3 = D_4\right]Q = \begin{bmatrix} 1 & 11 & x & x & 1 \\ 1 & 11 & x & x & 2 \\ 2 & 11 & y & y & 1 \\ 4 & 12 & x & x & 1 \\ 4 & 12 & x & x & 2 \end{bmatrix} = \begin{bmatrix} 1 & 11 & x & 1 \\ 1 & 11 & x & 2 \\ 2 & 11 & y & 1 \\ 4 & 12 & x & 1 \\ 4 & 12 & x & 2 \end{bmatrix}$$

Операция соединения отношений, наряду с операциями выборки и проекции, является одной из наиболее важных реляционных операций.

Обычно рассматривается несколько разновидностей операции соединения:

Общая операция соединения

Θ -соединение (тэта-соединение)

Экви-соединение

Естественное соединение

Наиболее важным из этих частных случаев является операция естественного соединения. Все разновидности соединения являются частными случаями общей операции соединения.

3.1 Общая операция соединения

Определение 3. Соединением отношений A и B по условию c называется отношение

(A TIMES B) WHERE c

с представляет собой логическое выражение, в которое могут входить атрибуты отношений A и B и (или) скалярные выражения.

Таким образом, операция соединения есть результат последовательного применения операций декартового произведения и выборки. Если в отношениях A и B имеются атрибуты с одинаковыми наименованиями, то перед выполнением соединения такие атрибуты необходимо переименовать.

3.2 Тэта-соединение

Определение 4. Пусть отношение A содержит атрибут X, отношение B содержит атрибут Y, а Θ - один из операторов сравнения ($=, \neq, <, \leq, >, \geq$ и т.д.). Тогда Θ -соединением отношения A по атрибуту X с отношением B по атрибуту Yназывают отношение

(A TIMES B) WHERE $X \Theta Y$

Это частный случай операции общего соединения.

Иногда, для операции Θ-соединения применяют следующий, более короткий синтаксис:

$A[X\Theta Y]B$

Пример 3. Рассмотрим некоторую компанию, в которой хранятся данные о поставщиках и поставляемых деталях. Пусть поставщикам и деталям присвоен некий статус. Пусть бизнес компании организован таким образом, что поставщики имеют право поставлять только те детали, статус которых не выше статуса поставщика (смысл этого может быть в том, что хороший поставщик с высоким статусом может поставлять больше разновидностей деталей, а плохой поставщик с низким статусом может поставлять только ограниченный список деталей, важность которых (статус детали) не очень высока).

Номер поставщика	Наименование поставщика	X (Статус поставщика)
1	Иванов	4
2	Петров	1
3	Сидоров	2

Таблица 5 Отношение A (Поставщики)

Номер детали	Наименование детали	Y (Статус детали)
1	Болт	3
2	Гайка	2
3	Винт	1

Таблица 6 Отношение B (Детали)

Ответ на вопрос "какие поставщики имеют право поставлять какие детали?" дает Θ-соединение $A[X \geq Y]B$:

Номер поставщика	Наименование поставщика	X (Статус поставщика)	Номер детали	Наименование детали	Y (Статус детали)
1	Иванов	4	1	Болт	3
1	Иванов	4	2	Гайка	2
1	Иванов	4	3	Винт	1
2	Петров	1	3	Винт	1
3	Сидоров	2	2	Гайка	2
3	Сидоров	2	3	Винт	1

Таблица 7 Отношение "Какие поставщики поставляют какие детали"

3.3 Экви-соединение

Наиболее важным частным случаем Θ-соединения является случай, когда Θ есть просто равенство.

Синтаксис экви-соединения:

$A[X = Y]B$

Пример 4. Пусть имеются отношения P, D и PD, хранящие информацию о поставщиках, деталях и поставках соответственно (для удобства введем краткие наименования атрибутов):

Номер поставщика PNUM	Наименование поставщика PNAME
1	Иванов
2	Петров
3	Сидоров

Таблица 8 Отношение P (Поставщики)

Номер детали DNUM	Наименование детали DNAME
1	Болт
2	Гайка
3	Винт

Таблица 9 Отношение D (Детали)

Номер поставщика PNUM	Номер детали DNUM	Поставляемое количество VOLUME
1	1	100
1	2	200
1	3	300
2	1	150
2	2	250
3	1	1000

Таблица 10 Отношение PD (Поставки)

Ответ на вопрос, какие детали поставляются поставщиками, дает экви-соединение $P[PNUM = PNUM]PD$. На самом деле, т.к. в отношениях имеются одинаковые атрибуты, то требуется сначала переименовать атрибуты, а потом выполнить экви-соединение. Запись становится более громоздкой:

$$(P \text{ RENAME } PNUM \text{ AS } PNUM1)[PNUM1 = PNUM2](PD \text{ RENAME } PNUM \text{ AS } PNUM2)$$

Обычно, такой сложной формой записи не пользуются. Но как бы то ни было, в результате имеем отношение:

Номер поставщика PNUM1	Наименование поставщика PNAME	Номер поставщика PNUM2	Номер детали DNUM	Поставляемое количество VOLUME
1	Иванов	1	1	100
1	Иванов	1	2	200
1	Иванов	1	3	300
2	Петров	2	1	150
2	Петров	2	2	250
3	Сидоров	3	1	1000

Таблица 11 Отношение "Какие детали поставляются какими поставщиками"

Недостатком экви-соединения является то, что если соединение происходит по атрибутам с одинаковыми наименованиями (а так чаще всего и происходит!), то в результатирующем отношении появляется два атрибута с одинаковыми значениями. В нашем примере атрибуты PNUM1 и PNUM2 содержат дублирующие данные. Избавиться от этого недостатка можно, взяв проекцию по всем атрибутам, кроме одного из дублирующих. Именно так действует естественное соединение.

3.4 Естественное соединение

Определение 5. Пусть даны отношения $A(A_1,A_2,\ldots,A_n,X_1,X_2,\ldots,X_p)$ и $B(X_1,X_2,\ldots,X_p,B_1,B_2,\ldots,B_m)$, имеющие одинаковые атрибуты $X_1,X_2,\ldots,X_p$ (т.е. атрибуты с одинаковыми именами и определенные на одинаковых доменах).

Тогда естественным соединением отношений A и B называется отношение с заголовком $(A_1,A_2,\ldots,A_n,X_1,X_2,\ldots,X_p,B_1,B_2,\ldots,B_m)$ и телом, содержащим множество кортежей $(a_1,a_2,\ldots a_n,x_1,x_2,\ldots,x_p,b_1,b_2,\ldots,b_m)$, таких, что $(a_1,a_2,\ldots a_n,x_1,x_2,\ldots,x_p)\in A$ и $(x_1,x_2,\ldots,x_p,b_1,b_2,\ldots b_m)\in B$.

Естественное соединение настолько важно, что для него используют специальный синтаксис:

A JOIN B

Замечание. В синтаксисе естественного соединения не указываются, по каким атрибутам производится соединение. Естественное соединение производится по всем одинаковым атрибутам.

Замечание. Естественное соединение эквивалентно следующей последовательности реляционных операций:

Переименовать одинаковые атрибуты в отношениях

Выполнить декартово произведение отношений

Выполнить выборку по совпадающим значениям атрибутов, имевших одинаковые имена

Выполнить проекцию, удалив повторяющиеся атрибуты

Переименовать атрибуты, вернув им первоначальные имена

Замечание. Можно выполнять последовательное естественное соединение нескольких отношений. Нетрудно проверить, что естественное соединение (как, впрочем, и соединение общего вида) обладает свойством ассоциативности, т.е.

(A JOIN B) JOIN C = A JOIN (B JOIN C)

поэтому такие соединения можно записывать, опуская скобки:

A JOIN B JOIN C

Пример 5. В предыдущем примере ответ на вопрос "какие детали поставляются поставщиками", более просто записывается в виде естественного соединения трех отношений P JOIN PD JOIN D (для удобства просмотра порядок атрибутов изменен, это является допустимым по свойствам отношений):

Номер поставщика PNUM	Наименование поставщика PNAME	Номер детали DNUM	Наименование детали DNAME	Поставляемое количество VOLUME
1	Иванов	1	Болт	100
1	Иванов	2	Гайка	200
1	Иванов	3	Винт	300
2	Петров	1	Болт	150
2	Петров	2	Гайка	250
3	Сидоров	1	Болт	1000

Таблица 12 Отношение P JOIN PD JOIN D

4 Деление

Пусть отношение R , называемое делимым, содержит атрибуты (A1,A2,...,An). Отношение S - делитель содержит подмножество атрибутов A: (A1,A2,...,Ak) (k<n). Результирующее отношение C определено на атрибутах отношения R, которых нет в S, т.е.

Ak+1,Ak+2,...,An. Кортежи включаются в результирующее отношение C только в том случае, если его декартово произведение с отношением S содержится в делимом R.

Деление / DIVISION /

Обозначение	Определение	LEAP :
$R_1\left[A_1 \div A_2\right]R_2$	$\left\{r\left[A_1\right]: r \in R_1 \wedge R_2\left[A_2\right] \subseteq g_R\left(r\left[A_1\right]\right)\right\}$	не поддерживается

Пример:

пусть

$$R_1\left(A_1, A_2\right) = \begin{bmatrix} a & x \\ a & y \\ a & z \\ b & x \\ c & y \end{bmatrix} \quad R_2\left(A_3, A_4\right) = \begin{bmatrix} 1 & x \\ 2 & x \\ 1 & y \end{bmatrix}$$

тогда

$$R_1\left[A_2 \div A_4\right] = \begin{bmatrix} a & x \\ a & y \\ a & z \\ b & x \\ c & y \end{bmatrix} \div \begin{bmatrix} x \\ y \end{bmatrix} = [a]$$

Определение 6. Пусть даны отношения $A(X_1,X_2,\ldots,X_n,Y_1,Y_2,\ldots,Y_m)$ и $B(Y_1,Y_2,\ldots,Y_m)$, причем атрибуты $Y_1,Y_2,\ldots,Y_m$ - общие для двух отношений. Делением отношений A на B называется отношение с заголовком $(X_1,X_2,\ldots,X_n)$ и телом, содержащим множество кортежей $(x_1,x_2,\ldots,x_n)$, таких, что для всех кортежей $(y_1,y_2,\ldots y_m)\in B$ в отношении A найдется кортеж $(x_1,x_2,\ldots,x_n,y_1,y_2,\ldots y_m)$.

Отношение A выступает в роли делимого, отношение B выступает в роли делителя. Деление отношений аналогично делению чисел с остатком.

Синтаксис операции деления:

A DEVIDEBY B

Замечание. Типичные запросы, реализуемые с помощью операции деления, обычно в своей формулировке имеют слово "все" - "какие поставщики поставляют все детали?".

Пример 6. В примере с поставщиками, деталями и поставками ответим на вопрос, "какие поставщики поставляют все детали?".

В качестве делимого возьмем проекцию X=PD[PNUM, DNUM], содержащую номера поставщиков и номера поставляемых ими деталей:

Номер поставщика PNUM	Номер детали DNUM
1	1
1	2
1	3
2	1
2	2
3	1

Таблица 13 Проекция X=PD[PNUM,DNUM]

В качестве делителя возьмем проекцию Y=[DNUM], содержащую список номеров всех деталей (не обязательно поставляемых кем-либо):

Номер детали DNUM

1
2
3

Таблица 14 Проекция Y=D[DNUM]

Деление *X* DEVIDEBY *Y* дает список номеров поставщиков, поставляющих все детали:

Номер поставщика PNUM
1

Таблица 15 Отношение X DEVIDEBY Y

Оказалось, что только поставщик с номером 1 поставляет все детали.

Отчет должен содержать:

1. Название, цель и задание практической работы;
2. Результат выполнения специальных реляционных операторов (полученные отношения);
3. Ответы на контрольные вопросы.

Контрольные вопросы:

1. Что такое «отношение», «домен», «кортеж»?
2. Какие существуют разновидности оператора соединения?
3. Какие реляционные операторы относятся к специальной группе?

Практическое занятие № 5

Тема: «Построение сложных запросов с учетом свойств теоретико-множественных операций и правил формальной оптимизации»

Цель практического занятия: освоить принципы работы теоретико-множественных операторов, научиться использовать операторы реляционной алгебры для выполнения сложных запросов.

Предварительная подготовка: лекционный материал по междисциплинарному курсу.

Количество часов: 2 часа

Постановка задачи

Используя синтаксис операторов реляционной алгебры, составить запрос в соответствии с вариантом задания.

Методика выполнения практической работы

Зависимые и примитивные реляционные операторы

Не все операторы реляционной алгебры являются независимыми - некоторые из них выражаются через другие реляционные операторы.

Такие реляционные операторы как, объединение, вычитание, декартово произведение, выборка, проекция, являются примитивными операторами - их нельзя выразить друг через друга.

Запросы, невыразимые средствами реляционной алгебры

Несмотря на мощь языка реляционной алгебры, имеется ряд типов запросов, которые принципиально нельзя выразить только при помощи операторов реляционной алгебры. Это вовсе не означает, что ответы на эти запросы нельзя получить вообще. Просто, для получения ответов на подобные запросы приходится применять процедурные расширения реляционных языков.

Примеры использования реляционных операторов

Пусть имеются отношения P, D и PD, хранящие информацию о поставщиках, деталях и поставках соответственно (для удобства введем краткие наименования атрибутов):

Номер поставщика *PNUM*	Наименование поставщика PNAME
1	Иванов
2	Петров
3	Сидоров

Таблица 1 Отношение P (Поставщики)

Номер детали *DNUM*	Наименование детали DNAME
1	Болт
2	Гайка
3	Винт

Таблица 2 Отношение D (Детали)

Номер поставщика *PNUM*	*Номер детали* *DNUM*	Поставляемое количество VOLUME
1	*1*	100
1	*2*	200

1	*3*	300
2	*1*	150
2	*2*	250
3	*1*	1000

Таблица 3 Отношение PD (Поставки)

Пример 1. Получить имена поставщиков, поставляющих деталь номер 2.
Решение:

$((DP \text{ JOIN } P) \text{ WHERE } DNUM = 2)[PNAME]$

Пример 2. Получить имена поставщиков, поставляющих по крайней мере одну гайку.
Решение:

$(((D \text{ WHERE } DNAME = Гайка) \text{ JOIN } DP) \text{ JOIN } P)[PNAME]$

Ответ на этот запрос можно получить и иначе:

$(((D \text{ JOIN } DP) \text{ JOIN } P) \text{ WHERE } DNAME = Гайка)[PNAME]$

Пример 3. Получить имена поставщиков, поставляющих все детали.
Решение:

$((DP[PNUM, DNUM] \text{ DEVIDEBY } D[DNUM]) \text{ JOIN } P)[PNAME]$

Пример 4. Получить имена поставщиков, не поставляющих деталь номер 2.
Решение:

$((P[PNUM] \text{ MINUS}((P \text{ JOIN } DP) \text{ WHERE } DNUM = 2)[PNUM]) \text{ JOIN } P)[PNAME]$

Ответ на этот запрос можно получить и пошагово:

$T_1 = P[PNUM]$ - получить список номеров всех поставщиков

$T_2 = P \text{ JOIN } DP$ - соединить данные о поставщиках и поставках

$T_3 = T_2 \text{ WHERE } DNUM = 2$ - в данных о поставщиках и поставках оставить только данные о поставках детали номер 2.

$T_4 = T_3[PNUM]$ - получить список номеров поставщиков, поставляющих деталь номер 2.

$T_5 = T_1 \text{ MINUS } T_4$ - получить список номеров поставщиков, не поставляющих деталь номер 2.

$T_6 = T_5 \text{ JOIN } P$ - соединить список номеров поставщиков, не поставляющих деталь номер 2 с данными о поставщиках (получатся полные данные о поставщиках, не поставляющих деталь номер 2).

Отчет должен содержать:

1. Название, цель и задание практической работы;
2. Результат выполнения реляционных операторов (решения в виде формул);
3. Ответы на контрольные вопросы.

Контрольные вопросы

1. Перечислить зависимые реляционные операторы, примитивные реляционные операторы.
2. Как решается проблема выполнения «невыполнимых» на языке реляционной алгебры запросов?
3. Почему реляционную алгебру называют «замкнутой»?
4. Какие два базовых механизма манипулирования реляционными данными существуют?

Практическое занятие № 6

Тема: «Обработка данных с помощью операции выборки SELECT»

Цель практического занятия освоить принципы работы операторов языка SQL, научиться использовать оператор SELECT для выполнения запросов к таблицам БД.

Предварительная подготовка: лекционный материал по междисциплинарному курсу.

Количество часов: 2 часа

Постановка задачи

Используя синтаксис оператора SELECT, составить запросы в соответствии с вариантом задания.

Структура таблиц

ЛАБОРАТОРИИ
(Код лаборатории: Текстовый,
Наименование лаборатории: Текстовый,
Код руководителя: Текстовый,
Дата организации лаборатории: Дата,
Дата закрытия лаборатории: Дата)

СПЕЦИАЛЬНОСТИ
(Код специальности: Текстовый,
Наименование специальности: Текстовый
Дата открытия специальности: Дата,
Дата закрытия специальности: Дата)

СПИСОК СЛУЖАЩИХ
(Табельный номер: Текстовый,
Фамилия: Текстовый,
Имя: Текстовый,
Отчество: Текстовый,
Пол: Текстовый (возможные значения М, Ж),
Семейное положение: (возможные значения Ж, Х, Р, З),
Код лаборатории: Текстовый,
Телефон: Текстовый,
Код специальности: Текстовый,
Оклад: Числовой,
День рождения: Дата,
Адрес: Текстовый,
Характеристика: Текстовый)

ПРЕМИИ
(Табельный номер: Текстовый,
Размер премии: Числовой,
Номер приказа: Текстовый,
Дата приказа: Дата)

ДЕТИ СОТРУДНИКОВ
(Табельный номер: Текстовый,
Фамилия ребенка: Текстовый,
Имя ребенка: Текстовый,
Дата рождения: Дата)

Варианты SQL-запросов

№ варианта	Задание
1	1. Список сотрудников, работающих в действующей лаборатории с минимальным размером фонда заработной платы по лаборатории. 2. Список всех служащих с максимального для сотрудника размера премии, если служащий не получал премий, то значение NULL.
2	1. Список руководителей действующих лабораторий с указанием числа служащих в лабораториях 2. Список сотрудников, работающих в действующих лабораториях, где число служащих превышает 10 человек.
3	1. Список сотрудников, работающих по специальностям, по которым число служащих не превышает 5 человек. 2. Список сотрудников, имеющих максимальный общий объем премий.
4	1. Создать запрос, позволяющий получить следующую информацию о сотруднике: ФИО, Дата рождения, Оклад, Надбавка (для родившихся до 1950 г. – 20% от оклада, после – 15% оклада). Данные упорядочить по полю

	Фамилия. 2. Список всех служащих с указанием количества детей, если служащий не имеет детей, то количество детей NULL.
5	1. Список сотрудников, работающих в действующей лаборатории, в которой наибольший размер средней заработной платы по лаборатории в целом. 2. Список руководителей лабораторий с указанием количества детей для каждого, если детей нет, то выводить NULL.
6	1. Список всех служащих с указанием размеров премий, получаемых ими, если служащий не получал премию ни разу, то размер его премии указать как NULL. 2. Список сотрудников, получающих оклад больше среднего по организации в целом.
7	1. Список лаборатории с указанием количества служащих в каждой. 2. Список действующих лабораторий с указанием объема премии, полученной каждой лабораторией.
8	1. Список руководителей лабораторий с указанием лаборатории. 2. Список лабораторий с указанием средней, максимальной и минимальной заработной платы по каждой лаборатории.
9	1. Найти самого молодого руководителя действующей лаборатории. 2. Найти самого молодого сотрудника, имеющего детей.
10	1. Найти сотрудника с максимальным объемом премии. 2. Список детей, у которых родители получают заработную плату ниже среднего по организации в целом.
11	1. Список сотрудников ни разу не получавших премии. 2. Список сотрудников имеющих более 3 детей и получающих заработную плату ниже среднего по организации в целом.
12	1. Создать запрос, позволяющий получить следующую информацию по детям: ФИО ребенка, Дата рождения, ФИО одного из родителей. Информацию выводить по детям, родившимся с 1990 по 2006 года. Данные упорядочить по полю Фамилии родителя. 2. Список разведенных служащих с указанием количества детей.
13	1. Список служащих с указанием суммарного размера премии сотрудника, полученного им за весь период работы, и отклонения суммарного размера премии сотрудника от максимального суммарного размера премии для сотрудников по организации в целом. 2. Список сотрудников, имеющих максимальный общий объем премий.

Методика выполнения практической работы

Для извлечения записей из таблиц в SQL определен оператор SELECT. С помощью этой команды осуществляется не только операция реляционной алгебры "выборка" (горизонтальное подмножество), но и предварительное соединение (join) двух и более таблиц. Это наиболее сложное и мощное средство SQL, полный синтаксис оператора SELECT имеет вид:

```
SELECT [ALL | DISTINCT] <список_выбора>
    FROM <имя_таблицы>, ...
    [ WHERE <условие> ]
    [ GROUP BY <имя_столбца>,... ]
      [ HAVING <условие> ]
    [ORDER BY <имя_столбца> [ASC | DESC],... ]
```

Порядок предложений в операторе SELECT должен строго соблюдаться (например, GROUP BY должно всегда предшествовать ORDER BY), иначе это приведет к появлению ошибок. Этот оператор всегда начинается с ключевого слова SELECT. В конструкции <список_выбора> определяется столбец или столбцы, включаемые в результат. Он может состоять из имен одного или нескольких столбцов, или из одного символа * (звездочка), определяющего все столбцы. Элементы списка разделяются запятыми.

Пример 1: получить список всех авторов

SELECT author FROM authors;

Пример 2: получить список всех полей таблицы authors:

SELECT * FROM authors;

В том случае, когда пользователя интересуют не все записи, а только те, которые удовлетворяют некому условию, это условие можно указать после ключевого слова WHERE.

Пример 3: найти все книги, опубликованные после 1996 года:

SELECT title FROM titles WHERE yearpub > 1996;

Пример 4: Допустим теперь, что надо найти все публикации за интервал 1995 - 1997 гг. Это условие можно записать в виде:

SELECT title FROM titles WHERE yearpub>=1995 AND yearpub<=1997;

Другой вариант этой команды можно получить с использованием логической операции проверки на вхождение в интервал:

SELECT title FROM titles WHERE yearpub BETWEEN 1995 AND 1997;

При использовании конструкции NOT BETWEEN находятся все строки, не входящие в указанный диапазон. Еще один вариант этой команды можно построить с помощью логической операции проверки на вхождение в список:

SELECT title FROM titles WHERE yearpub IN (1995,1996,1997);

Здесь задан в явном виде список интересующих значений. Конструкция NOT IN позволяет найти строки, не удовлетворяющие условиям, перечисленным в списке.

Некоторые задачи нельзя решить с использованием только операторов сравнения.

Пример 5: нужно найти web-site издательства "Wiley", но не известно его точного наименования. Для решения этой задачи предназначено ключевое слово LIKE, его синтаксис имеет вид:

WHERE <имя_столбца> LIKE <образец> [ESCAPE <ключевой_символ>]

Образец заключается в кавычки и должен содержать шаблон подстроки для поиска. Обычно в шаблонах используются два символа:

% (знак процента) - заменяет любое количество символов

_ (подчеркивание) - заменяет одиночный символ.

Попробуем найти искомый web-site:

SELECT publiser, url FROM publishers WHERE publisher LIKE '%Wiley%';

В соответствии с шаблоном СУБД найдет все строки включающие в себя подстроку "Wiley".

Пример 6: найти все книги, название которых начинается со слова "SQL":

SELECT title FROM titles WHERE title LIKE 'SQL%';

В том случае, когда надо найти значение, которое само содержит один из символов шаблона, используют ключевое слово ESCAPE и <ключевой_символ>. Литерал, следующий в шаблоне после ключевого символа, рассматривается как обычный символ, все последующие символы имеют обычное значение.

Пример 7: найти ссылку на web-страницу, о которой известно, что в ее url содержится подстрока "my_works":

SELECT site, url FROM wwwsites WHERE url LIKE '%my@_works%' ESCAPE '@';

Выборка из нескольких таблиц

Очень часто возникает ситуация, когда выборку данных надо производить из отношения, которое является результатом слияния (join) двух других отношений.

Пример 8: нужно получить из базы данных publications информацию о всех печатных изданиях в виде следующей таблицы:

|название_книги | год_выпуска | издательство |

Для этого СУБД предварительно должна выполнить слияние таблиц titles и publishers, а только затем произвести выборку из полученного отношения.

Для выполнения операции такого рода в операторе SELECT после ключевого слова FROM указывается список таблиц, по которым произвоится поиск данных. После ключевого слова WHERE указывается условие, по которому производится слияние. Для того, чтобы выполнить данный запрос, нужно дать команду:

```
SELECT titles.title,titles.yearpub,publishers.publisher
    FROM titles,publishers
    WHERE titles.pub_id=publishers.pub_id;
```

Пример 9: одновременно задаются условия и слияния, и выборки (результат предыдущего запроса ограничивается изданиями после 1996 года):

```
SELECT titles.title,titles.yearpub,publishers.publisher
    FROM titles,publishers
    WHERE titles.pub_id=publishers.pub_id AND
        titles.yearpub>1996;
```

Вычисления внутри SELECT

SQL позволяет выполнять различные арифметические операции над столбцами результирующего отношения. В конструкции <список_выбора> можно использовать константы, функции и их комбинации с арифметическими операциями и скобками.

Пример 10: чтобы узнать сколько лет прошло с 1992 года (год принятия стандарта SQL-92) до публикации той или иной книги можно выполнить команду:

```
SELECT title, yearpub-1992 FROM titles WHERE yearpub > 1992;
```

В арифметических выражениях допускаются операции сложения (+), вычитания (-), деления (/), умножения (*), а также различные функции (COS, SIN, ABS - абсолютное значение и т.д.). Также в запрос можно добавить строковую константу.

Пример 11:

```
SELECT 'the title of the book is', title, yearpub-1992
        FROM titles WHERE yearpub > 1992;
```

В SQL также определены так называемые агрегатные функции (таблица 1), которые совершают действия над совокупностью одинаковых полей в группе записей.

Таблица 1. Агрегирующие функции

Агрегирующая функция	Результат	Примечание
SUM([DISTINCT] выражение)	Сумма [различных] значений	только для числовых выражений, NULL значения игнорируются
AVG([DISTINCT] выражение)	Средняя величина [различных] значений	только для числовых выражений, NULL значения игнорируются
COUNT([DISTINCT] выражение)	Количество [различных] ненулевых значений	для всех типов выражений, NULL значения игнорируются
COUNT(*)	Количество выбранных строк	считают и NULL значения
MAX(выражение)	Максимальное значение	для всех типов выражений, NULL значения игнорируются
MIN(выражение)	Минимальное значение	для всех типов выражений, NULL значения игнорируются

Следует учитывать, что каждая агрегирующая функция возвращает единственное значение.

Пример 12: определить дату публикации самой "древней" книги в нашей базе данных

SELECT MIN(yearpub) FROM titles;

Пример 13: подсчитать количество книг в базе данных:

SELECT COUNT(*) FROM titles;

Область действия данных функции можно ограничить с помощью логического условия.

Пример 14: количество книг, в названии которых есть слово "SQL":

SELECT COUNT(*) FROM titles WHERE title LIKE '%SQL%';

Группировка данных

Группировка данных в операторе SELECT осуществляется с помощью ключевого слова GROUP BY и ключевого слова HAVING, с помощью которого задаются условия разбиения записей на группы.

GROUP BY неразрывно связано с агрегирующими функциями, без них оно практически не используется. GROUP BY разделяет таблицу на группы, а агрегирующая функция вычисляет для каждой из них итоговое значение.

Пример 15: определить количество книг каждого издательства в базе данных:

```
SELECT publishers.publisher, count(titles.title)
    FROM titles,publishers
    WHERE titles.pub_id=publishers.pub_id
    GROUP BY publisher;
```

Ключевое слово HAVING работает следующим образом: сначала GROUP BY разбивает строки на группы, затем на полученные наборы накладываются условия HAVING.

Пример 16: устранить из предыдущего запроса те издательства, которые имеют только одну книгу:

```
SELECT publishers.publisher, count(titles.title)
    FROM titles,publishers
    WHERE titles.pub_id=publishers.pub_id
    GROUP BY publisher
     HAVING COUNT(*)>1;
```

Сортировка данных

Для сортировки данных, получаемых при помощи оператора SELECT служит ключевое слово ORDER BY. С его помощью можно сортировать результаты по любому столбцу или выражению, указанному в <списке_выбора>. Данные могут быть упорядочены как по возрастанию, так и по убыванию.

Пример 17: сортировать список авторов по алфавиту: SELECT author FROM authors ORDER BY author;

Пример 18: получить список авторов, отсортированный по алфавиту, и список их публикаций, причем для каждого автора список книг сортируется по времени издания в обратном порядке (т.е. сначала более "свежие" книги, затем все более "древние"):

```
SELECT authors.author,titles.title,titles.yearpub,publishers.publisher
    FROM authors,titles,publishers,titleauthors
    WHERE titleauthors.au_id=authors.au_id AND
        titleauthors.title_id=titles.title_id AND
        titles.pub_id=publishers.pub_id
    ORDER BY authors.author ASC, titles.yearpub DESC;
```

Ключевое слово DESC задает здесь обратный порядок сортировки по полю yearpub, ключевое слов ASC (его можно опускать) - прямой порядок сортировки по полю author.

Отчет должен содержать:

1. Название, цель и задание практической работы;

2. Запрос, сформированный в соответствии с вариантом задания, с помощью оператора SELECT;

3. Ответы на контрольные вопросы.

Контрольные вопросы:

1. К какой категории языковых средств относится оператор SELECT?

2. Какие специальные символы используются в операторе SELECT для выполнения групповых операций?

3. Зачем в операторе SELECT использовать обращение к полям по составному имени?

Практическое занятие № 7

Тема: «Редактирование базы данных с помощью SQL-инструкций UPDATE, INSERT, DELETE »

Цель практического занятия: освоить принципы работы операторов языка SQL, научиться использовать операторы манипулирования данными для редактирования таблиц БД.

Количество часов: 2 часа

Постановка задачи

Используя синтаксис операторов манипуляции данных INSERT, DELETE, UPDATE, составить команды в соответствии с вариантом задания.

Структура таблиц

ЛАБОРАТОРИИ
(Код лаборатории: Текстовый,
Наименование лаборатории: Текстовый,
Код руководителя: Текстовый,
Дата организации лаборатории: Дата,
Дата закрытия лаборатории: Дата)

СПЕЦИАЛЬНОСТИ
(Код специальности: Текстовый,
Наименование специальности: Текстовый
Дата открытия специальности: Дата,
Дата закрытия специальности: Дата)

ПРЕМИИ
(Табельный номер: Текстовый,
Размер премии: Числовой,
Номер приказа: Текстовый,
Дата приказа: Дата)

ДЕТИ СОТРУДНИКОВ
(Табельный номер: Текстовый,
Фамилия ребенка: Текстовый,
Имя ребенка: Текстовый,
Дата рождения: Дата)

СПИСОК СЛУЖАЩИХ
(Табельный номер: Текстовый,
Фамилия: Текстовый,
Имя: Текстовый,
Отчество: Текстовый,
Пол: Текстовый (возможные значения М, Ж),
Семейное положение: (возможные значения Ж, Х, Р, З),
Код лаборатории: Текстовый,
Телефон: Текстовый,
Код специальности: Текстовый,
Оклад: Числовой,
День рождения: Дата,
Адрес: Текстовый,
Характеристика: Текстовый)

Варианты заданий

№ варианта	Задание
1	1. Удалить из списка сотрудников с минимальным размером фонда заработной платы. 2. Изменить данные максимального размера премии на значение NULL. 3. Добавить лабораторию со следующими данными; код лаборатории – 09406, наименование лаборатории «Лаборатория инфокоммуникационных систем», код руководителя – 234, дата основания – 30.01.2014, дата закрытия – пробел.
2	1. Обновить список детей сотрудника с табельным номером 2301, вписав значение Агапкин Сергей, 29.01.2014» 2. Удалить из списка лаборатории, дата организации которых позже 2000 г. 3. Добавить для сотрудника с табельным номером 3405 размер премии 500 у.е., по приказу №20145 от 22.01.2014.
3	1. Удалить из списка сотрудников, старше 65 лет. 2. Обновить данные поля «Семейное положение» списка сотрудников, изменив значение «З» на «Ж». 3. Добавить лабораторию со следующими данными; код лаборатории – 04086, наименование лаборатории «Лаборатория информационных технологий», код руководителя – 234, дата основания – 30.01.2002, дата закрытия – 1.01.2014.
4	1. Добавить свои личные данные в список сотрудников. 2. Удалить из списка детей сотрудников данные, имеющие значение «Null» 3. Обновить данные лаборатории с кодом – 04086, изменив наименование лаборатории «Лаборатория информационных технологий» на новое - «Лаборатория информационных систем».

Методика выполнения практической работы

Язык манипулирования данными (DML)

DML - Data Manipulation Language -язык манипулирования данными, спользуется для работы с информацией, хранимой в базе данных.

Основными командами этой группы являются:

Select - выборка информации.

Insert - добавление информации.

Update - обновление информации.

Delete - удаление информации.

Добавление новой записи в таблицу INSERT

INSERT INTO <имя_таблицы> [(<имя_столбца>,<имя_столбца>,...)]

VALUES (<значение>,<значение>,..)

Список столбцов в данной команде не является обязательным параметром. В этом случае должны быть указаны значения для всех полей таблицы в том порядке, как эти столбцы были перечислены в команде CREATE TABLE.

Пример 1:

INSERT INTO publishers VALUES (16,"Microsoft Press","http://www.microsoft.com");

Пример 2 с указанием списка столбцов:

INSERT INTO publishers (publisher,pub_id) VALUES ("Super Computer Publishing",17);

Пример 3:

INSERT INTO medium_type_directory (medium_type) VALUES ('dvd')

В таблицу medium_type_directory добавлена новая запись.

Модификация записей UPDATE

UPDATE <имя_таблицы> SET <имя_столбца>=<значение>,...

[WHERE <условие>]

Если задано ключевое слово WHERE и условие, то команда UPDATE применяется только к тем записям, для которых оно выполняется. Если условие не задано, UPDATE применяется ко всем записям.

Пример 4:

UPDATE publishers SET url="http://www.superpub.com" WHERE pub_id=17;

В качестве условия используются логические выражения над константами и полями. В условиях допускаются: операции сравнения: > , < , >= , <= , = , <> , != . В SQL эти операции могут применяться не только к числовым значениям, но и к строкам ("<" означает раньше, а ">" позже в алфавитном порядке) и датам ("<" раньше и ">" позже в хронологическом порядке).

операции проверки поля на значение NULL: IS NULL, IS NOT NULL

операции проверки на вхождение в диапазон: BETWEEN и NOT BETWEEN.

операции проверки на вхождение в список: IN и NOT IN

операции проверки на вхождение подстроки: LIKE и NOT LIKE

отдельные операции соединяются связями AND, OR, NOT и группируются с помощью скобок.

Пример 5: UPDATE publishers SET url="url not defined" WHERE url IS NULL;

Эта команда находит в таблице publishers все неопределенные значения столбца url и заменяет их строкой "url not defined".

Пример 6:

UPDATE clients

SET client_phone_number=’795-55-78-48’

WHERE client_fio=’John N. Doe’

В данном примере у клиента John N. Doe будет изменен номер телефона.

Удаление записей DELETE

DELETE FROM <имя_таблицы> [WHERE <условие>]

Удаляются все записи, удовлетворяющие указанному условию. Если ключевое слово WHERE и условие отсутствуют, из таблицы удаляются все записи.

Пример 7: DELETE FROM publishers WHERE publisher = "Super Computer Publishing";

Эта команда удаляет запись об издательстве Super Computer Publishing.

Пример 8:

DELETE FROM clients WHERE client_id=15

В примере выполняется удаление записи о клиенте с client_id равным 15.

Отчет должен содержать:

1. Название, цель и задание практической работы;
2. Команды удаления, вставки, модификации записей в соответствии с вариантом задания;
3. Ответы на контрольные вопросы.

Контрольные вопросы:

1. К какой категории языковых средств относятся операторы удаления, вставки, обновления записей?
2. Какие из используемых команд позволяют выполнять операции в соответствии с накладываемым ограничением (условием)?
3. Какие логические выражения используются в операторах для формирования условия?

Практическое занятие № 8

Тема: «Создание структуры базы данных с помощью SQL-инструкций»

Цель практического занятия: освоить принципы работы операторов языка SQL, научиться использовать операторы определения структуры БД.

Количество часов: 2 часа.

Постановка задачи

Используя синтаксис операторов определения структуры БД, составить команды в соответствии с вариантом задания.

Структура таблиц

ЛАБОРАТОРИИ
(Код лаборатории: Текстовый,
Наименование лаборатории: Текстовый,
Код руководителя: Текстовый,
Дата организации лаборатории: Дата,
Дата закрытия лаборатории: Дата)

СПИСОК СЛУЖАЩИХ
(Табельный номер: Текстовый,
Фамилия: Текстовый,
Имя: Текстовый,
Отчество: Текстовый,
Пол: Текстовый (возможные значения М, Ж),
Семейное положение: (возможные значения Ж, Х, Р, З),
Код лаборатории: Текстовый,
Телефон: Текстовый,
Код специальности: Текстовый,
Оклад: Числовой,
День рождения: Дата,
Адрес: Текстовый,
Характеристика: Текстовый)

СПЕЦИАЛЬНОСТИ
(Код специальности: Текстовый,
Наименование специальности: Текстовый
Дата открытия специальности: Дата,
Дата закрытия специальности: Дата)

ПРЕМИИ
(Табельный номер: Текстовый,
Размер премии: Числовой,
Номер приказа: Текстовый,
Дата приказа: Дата)

ДЕТИ СОТРУДНИКОВ
(Табельный номер: Текстовый,
Фамилия ребенка: Текстовый,
Имя ребенка: Текстовый,
Дата рождения: Дата)

Варианты заданий

№ варианта	Задание
1	Удалить из списка сотрудников с минимальным размером фонда заработной платы. Изменить данные максимального размера премии на значение NULL. Добавить лабораторию со следующими данными; код лаборатории – 09406, наименование лаборатории «Лаборатория инфокоммуникационных систем», код руководителя – 234, дата основания – 30.01.2014, дата закрытия – пробел.
2	Обновить список детей сотрудника с табельным номером 2301, вписав значение Агапкин Сергей, 29.01.2014» Удалить из списка лаборатории, дата организации которых позже 2000 г. Добавить для сотрудника с табельным номером 3405 размер премии 500 у.е., по приказу №20145 от 22.01.2014.

Методика выполнения практической работы

Определение структур базы данных (DDL)

Язык определения данных (DDL) является частью SQL, дающей пользователю возможность создавать различные объекты базы данных и переопределять их структуру, например, создавать или удалять таблицы.

Среди основных команд DDL: CREATE TABLE, ALTER TABLE, DROP TABLE, CREATE INDEX, DROP INDEX

Команда создания таблицы CREATE

Таблицы создаются командой CREATE TABLE. Эта команда создает пустую таблицу — таблицу без строк. Значения вводятся с помощью DML команды INSERT. Команда CREATE TABLE в основном определяет имя таблицы, в виде описания набора имен столбцов указанных в определенном порядке. Она также определяет типы данных и размеры столбцов. Каждая таблица должна иметь, по крайней мере, один столбец. Синтаксис команды CREATE TABLE:

CREATE TABLE <table-name >
(<column name > <data type>[(<size>)],
<column name > <data type> [(<size>)] ...);

Эта команда будет создавать таблицу Продавцов:

Пример 1.

```
CREATE TABLE Saleepeople
(snum  integer,
 sname char (10),
 city  char (10));
```

Пример 2: эта инструкция присваивает новой таблице имя FILMS и определяет для каждого ее столбца имя и тип данных, хранимых в нем.

```
CREATE TABLE films (
        film_id INTEGER NOT NULL,
        film_name VARCHAR(100) NOT NULL,
        film_time time,
        film_director VARCHAR(50) NOT NULL,
        film_actors VARCHAR(255),
        film_year INTEGER NOT NULL,
        PRIMARY KEY (film_id))
```

Порядок столбцов в таблице определяется порядком, в котором они указаны. Имя столбца не должно разделяться при переносе строки, но отделяется запятыми.

Изменение существующей таблицы ALTER TABLE

Команда ALTER TABLE не часть стандарта ANSI; но это — широко доступная, и довольно содержательная форма, хотя ее возможности несколько ограничены. Она используется, чтобы изменить определение существующей таблицы. Обычно, она добавляет столбцы к таблице. Иногда она может удалять столбцы или изменять их размеры, а также в некоторых программах добавлять или удалять ограничения. Типичный синтаксис, чтобы добавить столбец к таблице:

ALTER TABLE <table name> ADD/DROP <column name> <data type> <size>;

Пример 3.

ALTER TABLE film_distributions ADD FOREIGN KEY (film_id) REFERENCES films(film_id) ON DELETE CASCADE

Удаление таблиц DROP

Вы должны быть собственником (т.е. быть создателем) таблицы, чтобы иметь возможность удалить ее. Поэтому не волнуйтесь о случайном разрушении ваших данных, SQL сначала потребует, чтобы вы очистили таблицу прежде, чем удалит ее из базы данных. Таблица с находящимися в ней строками, не может быть удалена.

DROP TABLE <table name>;

Пример 4.

DROP TABLE films

При подаче этой команды, имя таблицы больше не распознается, и нет такой команды, которая могла бы быть дана этому объекту. Вы должны убедиться, что эта таблица не ссылается внешним ключом к другой таблице, и что она не используется в определении Представления.

Примечание. Не все SQL-серверы требуют очистки таблицы перед ее удалением. Здесь нужно обратиться к документации по системе.

Индексы

Индекс — это упорядоченный (буквенный или числовой) список столбцов или групп столбцов в таблице. Таблицы могут иметь большое количество строк, а, так как строки не находятся в каком-нибудь определенном порядке, их поиск по указанному значению может потребовать значительного времени. Когда вы создаете индекс в поле, ваша база данных запоминает соответствующий порядок всех значений этого поля в области памяти. Предположим, что наша таблица Заказчиков имеет тысячи входов, а вы хотите найти заказчика с номером cnum=299. Так как строки не упорядочены, ваша программа будет просматривать всю таблицу, строку за строкой, проверяя каждый раз значение поля cnum на равенство значению 299. Однако если бы имелся индекс в поле cnum, то программа могла бы выйти на номер 299 прямо по индексу и дать информацию о том, как найти правильную строку таблицы.

Создание индекса

CREATE INDEX <index name> ON <table name>

(<column name> [,<column name>]...);

Таблица, конечно, должна уже быть создана и должна содержать имя столбца. Имя индекса не может быть использовано для чего-то другого в базе данных (любым пользователем). Однажды созданный, индекс будет невидим пользователю. Сервер SQL сам решает, когда он необходим, чтобы ссылаться на него, и делает это автоматически. Если, например, таблица Заказчиков будет наиболее часто упоминаемой в запросах продавцов к их собственной клиентуре, было бы правильно создать такой индекс в поле snum таблицы Заказчиков.

Пример 5.

CREATE INDEX Clientgroup ON Customers (snum);

Теперь, тот продавец, который имеет отношение к этой таблице, сможет найти собственную клиентуру очень быстро.

Удаление индекса

Главным признаком индекса является его имя, поэтому он может быть удален. Обычно пользователи не знают о существовании индекса. SQL автоматически определяет, позволено ли пользователю использовать индекс, и если да, то разрешает использовать его. Однако, если вы хотите удалить индекс, вы должны знать его имя. Этот синтаксис используется для удаления индекса:

DROP INDEX <Index name>;

Пример 6.

CREATE INDEX Clientgroup

Удаление индекса не воздействует на содержание полей.

Отчет должен содержать:

1. Название, цель и задание практической работы;
2. Команды для определения структуры базы данных;
3. Ответы на контрольные вопросы.

Контрольные вопросы:

1. Какие команды служат для определения структуры базы данных (DDL)?
2. Синтаксис команды CREATE.
3. Синтаксис команды ALTER.