Министерство образования Республики Беларусь

Учреждение образования

Белорусский государственный университет информатики и радиоэлектроники

Факультет непрерывного и дистанционного обучения

Кафедра информатики

Электронный учебно-методический комплекс

по дисциплине

**Модели данных**

**и**

**системы управления базами данных**

**(часть 1)**

Для студентов специальности

**1-31 03 04 “Информатика“**

Минск 2011

# Общие сведения

## Сведения об ЭУМК

Данный комплекс специально разработан для студентов, обучающихся дистанционно. Целью его создания было сделать процесс изучения данной дисциплины максимально удобным и комфортным для студентов. Кроме того, обладание комплексом должно позволить студенту минимизировать свои непосредственные контакты с университетом и компьютером, что особенно удобно для иногородних студентов, а также тех, кто по различным причинам вынужден на долгое время уезжать из города. В принципе данный комплекс позволяет студенту изучить дисциплину и подготовиться к сдаче экзамена по ней “автономно”

Выше приведён перечень и описание составных частей данного комплекса. Они выполнены в виде гиперссылок и для перехода к нужной части требуется, удерживая клавишу “Ctrl”, щёлкнуть левой кнопкой мыши выбранную гиперссылку. Впрочем, поскольку весь комплекс представляет собой файл Microsoft Word, его можно просматривать в обычном режиме, а также полностью или частично печатать

**Составитель: Калоша Е.П., Тараканов А.Н.**, доцент и ассистент**,** кафедры информатики Учреждения образования «Белорусский государственный университет информатики и радиоэлектроники».

Рассмотрен и рекомендован к изданию на заседании кафедры информатики, протокол № \_\_ от \_\_.\_\_.2011.

## Методические рекомендации по изучению дисциплины

В соответствии с учебным планом студенты дистанционной формы обучения инженерных специальностей изучают курс «Модели данных и системы управления базами данных (часть 1)».

Учебным планом по данному курсу предусмотрено изучение теоретических вопросов, решение задач, выполнение 2 контрольных работ и 2 ИПР с ИКТ. Изучение курса заканчивается сдачей экзамена. К сдаче экзамена студенты допускаются только при условии выполненных и защищенных контрольных работ и ИПР с ИКТ.

Рекомендуется изучать курс «Модели данных и системы управления базами данных (часть 1)» в соответствии с рабочей программой. Сначала необходимо ознакомиться с содержанием курса, затем изучить рекомендуемую литературу, обращая внимание на вопросы, выделенные в рабочей программе, после чего изучить теоретическое изложение курса по приведенным разделам, темам и вопросам, ответить на контрольные вопросы, выполнить задачи для решения (выполнения контрольных работ) в соответствии с заданием.

Так как теоретический материал излагается в строгой логической последовательности, рекомендуется изучать данную дисциплину, придерживаясь данной логики.

## Рабочая учебная программа

**Учреждение образования**

**«Белорусский государственный университет**

**информатики и радиоэлектроники»**

УТВЕРЖДАЮ

Декан факультета непрерывного и дистанционного обучения

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. М. Бондарик

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2010 г.

Регистрационный № -\_\_\_/.

**Модели данных**

**и**

**системы управления базами данных (часть 1)**

Рабочая учебная программа

**для направления специальности 1-31 03 04**

**“Информатика“**

Факультет **непрерывного и дистанционного обучения**

Кафедра **информатики**

Курс **пятый (семестр 9)**

**Контрольные работы** **2 работы**

**ИПР с ИКТ 2 работы**

Всего часов **93 часа**

Зачет **5 курс**

Форма получения

высшего образования **дистанционная**

Минск 2011

Составители Калоша Е.П., Тараканов А.Н.

Учебная программа составлена на основе типовой учебной  *«Модели данных и системы управления базами данных (часть 1)*» для специальности 1-31 03 04 Информатика, утвержденной Министерством образования Республики Беларусь 14. 04. 2010 регистрационный № ТД-G.266/тип.

Рассмотрена и рекомендована к утверждению на заседании кафедры информатики

протокол № \_\_ от \_\_\_\_\_ \_\_\_\_

Заведующий кафедрой Минченко Л.И.

Одобрена и рекомендована к утверждению Научно-методической комиссией факультета компьютерных систем и сетей Учреждения образования «Белорусский государственный университет информатики и радиоэлектроники»

протокол № \_\_ от \_\_\_\_\_ \_\_\_

Председатель Лукашевич М.М.

СОГЛАСОВАНО

Начальник отделаметодического обеспечения

учебного процесса \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ц. С. Шикова

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Цель преподавания дисциплины. Дисциплина «Модели данных и системы управления базами данных (часть 1)» знакомит студентов с теоретическими основами БД и методами решения практических задач в этой области. Она связана с дисциплинами «Объектно-ориентированное программирование», «Теория алгоритмов и методы трансляции», «Методы и технологии программирования».

**Задачи изучения дисциплины.**В процессе обучения, студенты должны изучить методы проектирования баз данных; познакомиться с реляционной моделью, изучить языки запросов к базам данных; усвоить связь между теоретическими положениями, лежащими в основе баз данных, и практическими задачами проектирования, разработки СУБД и прикладных программ управления БД; приобрести навыки к решению практических задач проектирования баз данных и разработки программных приложений в этой области.

2. ПЕРЕЧЕНЬ ТЕМ ЛАБОРАТОРНЫХ ЗАНЯТИЙ, ИХ СОДЕРЖАНИЕ И ОБЪЕМ В ЧАСАХ

Лабораторные работыучебным планом не предусмотрены

Содержание дисциплины.

1. Название тем теоретического материала, их содержание, рекомендуемый объем в часах

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ку  р  с | | Название и содержание тем (*по типовой или учебной программе*) | Контрольная работа (номер и тема по п.2) | Лабораторная работа с указанием вида 1  (по п.1) | Оснащение контрольных и лабораторных работ  (по п.5) | Литература (по п.4) | Рекомендуемый объем для изучения (в часах)2 | Форма контроля знаний (зачет по контрольной работе, тесты, защита лабораторной работы, защита курсового проекта, экзамен, зачет) |
| 1 | 2 | | 3 | 4 | 5 | 6 | 7 | 8 |
| **Часть 2** | | | | | | | | |
|  | | ***Тема 1. Системы баз данных***  Достоинства СБД – архитектура СБД - компоненты системы баз данных |  |  |  |  | 3 |  |
|  | | ***Тема 2. Структурные диаграммы***  Метод структурных диаграмм – ER-диаграммы – стандарты проектирования БД | КР №1 |  |  |  | 3 | Зачет по контрольной работе №1 |
|  | | ***Тема 3. Сетевая модель данных***  Модели данных – сетевая модель – преобразование структурной диаграммы в сетевую модель – сетевая модель на физическом уровне – пример реализации сетевой модели – оценка сетевой модели |  |  |  |  | 3 |  |
|  | | ***Тема 4. Иерархическая модель данных***  Иерархическая модель – преобразование структурной диаграммы в иерархическую модель – иерархическая модель на физическом уровне – пример реализации иерархической модели – оценка иерархической модели | КР №2 |  |  |  | 3 | Зачет по контрольной работе №2 |
|  | | ***Тема 5. Реляционная модель данных***  Реляционные объекты данных – целостность реляционной модели – преобразование структурных диаграмм в реляционную модель |  |  |  |  | 2 |  |
|  | | ***Тема 6. Операции с реляционными объектами***  Реляционная алгебра – описание операций – пример реализации |  | ИПР с ИКТ№1 |  |  | 2 | Защита ИПР с ИКТ №1 |
|  | | ***Тема 7. Реляционные исчисления***  Основные положения – исчисление кортежей – исчисление доменов – пример реализации |  |  |  |  | 2 |  |
|  | | ***Тема 8. SQL***  1. История – язык определения данных – реализация реляционных правил целостности в языке SQL  2. Язык манипуляции данными – оператор SELECT – необходимые секции SELECT  3. Выборка и сортировка данных – секция WHERE – секция ORDER BY  4. Выборка из нескольких таблиц – секция JOIN  5. Функции в SQL – простейшие функции – агрегатные функции в SQL – средства группировки  6. Подзапросы – синтаксис подзапросов – подзапросы по нескольким столбцам  7. Язык манипуляции данными – операторы INSERT, DELETE, UPDATE |  | ИПР с ИКТ№2 |  |  | 14 | Защита ИПР с ИКТ №2 |

2.КОНТРОЛЬНЫЕ РАБОТЫ, ИХ ХАРАКТЕРИСТИКИ

|  |  |  |  |
| --- | --- | --- | --- |
| № пп | Тема | Характеристика | Рекомендуемый объем  в часах |
| 1. | 1.1 Основы SQL\*Plus  1.2 SQL: простейшие запросы | 1.1 Ознакомиться с возможностями среды SQL\*Plus.  Ознакомиться с основными командами среды SQL\*Plus.  ИспользоватьSQL\*Plus для выполнения и редактирования запросов SQL.  1.2 Изучить структуры запроса SELECT. Получить практические навыки в использовании следующих базовых возможностей:  1.2.1выборка данных из одной таблицы;  1.2.2использование арифметических выражений и работа с символьными значениями в запросах;  1.2.3сортировка результатов запроса. | 8 |
| 2. | 2.1 SQL: DDL  2.2 SQL: INSERT / UPDATE / DELETE | 2.1 Изучить основные возможности Data Definition Language (DDL). Получить практические навыки выполнения с его помощью, следующих операций:  2.1.1создание и удаление таблиц,  2.1.2 изменение таблиц.  2.2 Получить практические навыки вставки/удаления/модификации данных в таблицах с помощью соответствующих операторов DataManipulationLanguage. | 8 |

3.ИНДИВИДУАЛЬНЫЕ ПРАКТИЧЕСКИЕ РАБОТЫ, ИХ ХАРАКТЕРИСТИКИ

|  |  |  |  |
| --- | --- | --- | --- |
| № пп | Название темы | Содержание | Объем в часах |
| 1 | 2 | 3 | 4 |
| Девятый семестр | | | |
| 1. | 1.1 SQL: WHERE, JOIN  1.2 SQL\*Plusформатирование вывода, переменные подстановки | 1.1 Построить выборки с использованием секции WHERE оператора SELECT. Построить выборки из нескольких таблиц с использованием различных приемов объединения таблиц.  1.2 Получить данные от пользователя в процессе выполнения командных скриптов и реализации интерактивных запросов.Изучить возможностиSQL\*Plus для форматирования вывода результатов запроса и построения простейших отчетов. | 8 |
| 2. | 2.1 SQL: функции  2.2 SQL: группировки | 2.1 Изучить различные группы встроенных однострочных функций в SQLи получить навыки их использования.  2.2 Группировать данные выборки с помощью секции GROUPBY оператора SELECT. Изучить средства последующего ограничения выборки с помощью секции HAVING. | 8 |
| Итого: | | | 16 |

4.КУРСОВЫЕ РАБОТЫ (ПРОЕКТЫ), ИХ ХАРАКТЕРИСТИКИ

Курсовые работы учебным планом не предусмотрены

5. ЛИТЕРАТУРА

5.1.ОСНОВНАЯ

* + - 1. Дейт К.Дж. Введение в системы баз данных. К.; Диалектика, 1998
      2. Гарсиа-Молина Г., Ульман Дж., Уидом Дж. Системы баз данных. М.: Издательский дом «Вильямс», 2003
      3. Мейер Д. Теория реляционных баз данных. М. Мир. 1987

# Теоретический раздел

## Лекции

### 1. Модели данных реляционная модель.

**1.1** СБД предоставляет пользователю некие стандартные способы организации данных и методы доступа к ним. Эти стандартные способы задаются понятием модели данных.

Модель данных связана с тремя аспектами данных:

1.структурой;

2.целостностью;

3.обработкой данных.

**1.2**Структуры данных описывают объекты, в которых хранятся сущности и отношения.

Правила целостности описывают определённые законы, которые должны соблюдаться в БД, если она соответствует модели.

Операции над данными позволяют представить запросы в БД и манипуляции над данными.

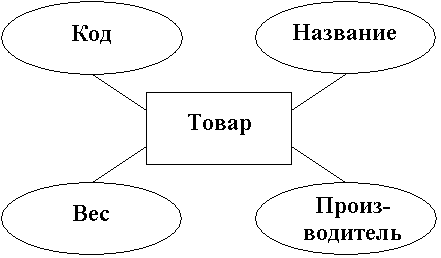
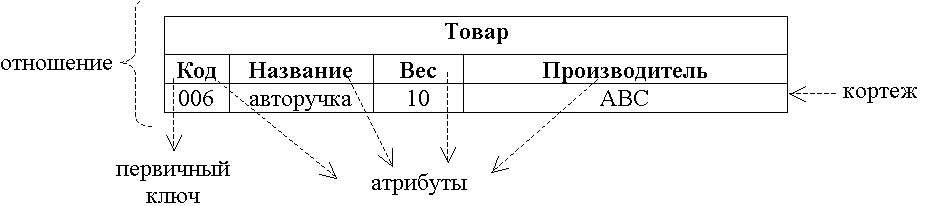
Существует три основных модели данных:

• иерархическая;

• сетевая;

• реляционная.

Между структурной диаграммой и реляционной моделью имеется следующее соответствие:



Домен-множество значений, которые принимает атрибут.

Количество атрибутов - степень (арность) отношения.

Первичный ключ - атрибут или совокупность атрибутов таких, что в любой момент времени не существует двух или более кортежей, содержащих одинаковые значения в этих атрибутах.

Переменная "отношение именованный объект, значение которого является отношением.

Реляционная БД - БД, воспринимаемая пользователем как набор отношений.

**1.4 Целостность реляционной БД**

в любой БД существуют правила целостности.

1. Потенциальные ключи.

Пусть R - переменная отношения, тогда потенциальный ключ К для R-подмножеств множества атрибутов R, всегда обладает следующими свойствами:

• уникальности (нет двух различных кортежей в некотором значении R с одинаковыми значениями К);

• не избыточности (никакое из подмножеств К не обладает свойством уникальности).

В реляционной модели один из потенциальных ключей выбирается в качестве первичного, а остальные называются альтернативными.

1. Внешние ключи.



Пусть R2 - отношение, тогда внешний ключ FK (Foreign кеу) в отношении R2 - это подмножество множества атрибутов в R2 такое, что:

• существует базовое отношение R1 с потенциальным ключом СК;

• каждое значение FK в текущем значении R2 всегда совпадает со значением СК некоторого кортежа в текущем значении R1.

Каскадирование.

### 2. Целостность реляционной модели.

СБД предоставляет пользователю некие стандартные способы организации данных и методы доступа к ним. Эти стандартные способы задаются понятием модели данных.

Модель данных связана с тремя аспектами данных:

1.структурой;

2.целостностью;

3.обработкой данных.

в любой БД существуют правила целостности. Они либо зависят от приложения (на- пример номера месяцев от 1 до 12), либо должны выполняться всегда для каждой БД.

**1. Потенциальные ключи.**

Пусть R - переменная отношения, тогда потенциальный ключ К для R-подмножеств множества атрибутов R, всегда обладает следующими свойствами:

• уникальности (нет двух различных кортежей в некотором значении R с одинаковыми значениями К);

• не избыточности (никакое из подмножеств К не обладает свойством уникальности).

В реляционной модели один из потенциальных ключей выбирается в качестве первичного, а остальные называются альтернативными.

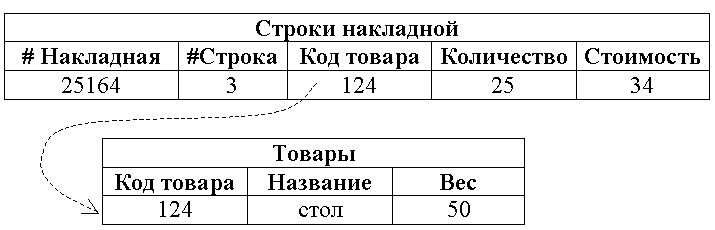
**2.Внешние ключи.**

Пусть R2 - отношение, тогда внешний ключ FK (Foreign кеу) в отношении R2 - это подмножество множества атрибутов в R2 такое, что:

• существует базовое отношение R1 с потенциальным ключом СК;

• каждое значение FK в текущем значении R2 всегда совпадает со значением СК некоторого кортежа в текущем значении R1.

БД должна обладать ссылочной целостностью, т.е. внешние ключи не должны содержать неверные значения, т.е. внешние ключи всегда должны быть согласованы, т.е. значению внешнего ключа в кортеже ссылающегося отношения должен всегда соответствовать кортеж; в целевом отношении.

Пример: как обеспечить ссылочную целостность?

Если удалить товар с кодом 124, либо изменить его в отношении Товары, то ссылочная целостность будет нарушена. Т.о. СУБД должна решать две следующие проблемы:

1. Что случится при попытке удалить объект ссылки внешнего ключа?

• ограничить (удалить нельзя) - запрет удалять целевой кортеж;пока есть хотя бы один ссылающийся на него кортеж;;

• каскадировать - удалить целевой кортеж; и все ссылающиеся.

2. Что должно случиться при попытке модифицировать потенциальный ключ, на который ссылается внешний ключ?

• ограничить;

• каскадировать.

При задании схемы БД разработчик должен указать не только множество атрибутов, составляющих потенциальный и внешний ключи, но и определить поведение СУБД при вышеперечисленных ситуациях.

1. Целостность атрибута. Значение атрибута должно быть из соответствующего ему домена.

### 3. Ссылочная целостность внешних ключей в реляционной модели.

СБД предоставляет пользователю некие стандартные способы организации данных и методы доступа к ним. Эти стандартные способы задаются понятием модели данных.

Модель данных связана с тремя аспектами данных:

1.структурой;

2.целостностью;

3.обработкой данных.

в любой БД существуют правила целостности. Они либо зависят от приложения (на- пример номера месяцев от 1 до 12), либо должны выполняться всегда для каждой БД.

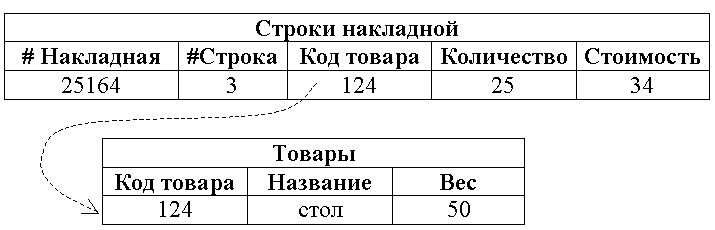
**2.Внешние ключи.**

Пусть R2 - отношение, тогда внешний ключ FK (Foreign кеу) в отношении R2 - это подмножество множества атрибутов в R2 такое, что:

• существует базовое отношение R1 с потенциальным ключом СК;

• каждое значение FK в текущем значении R2 всегда совпадает со значением СК некоторого кортежа в текущем значении R1.

БД должна обладать ссылочной целостностью, т.е. внешние ключи не должны содержать неверные значения, т.е. внешние ключи всегда должны быть согласованы, т.е. значению внешнего ключа в кортеже ссылающегося отношения должен всегда соответствовать кортеж; в целевом отношении.

Пример: как обеспечить ссылочную целостность?

Если удалить товар с кодом 124, либо изменить его в отношении Товары, то ссылочная целостность будет нарушена. Т.о. СУБД должна решать две следующие проблемы:

1. Что случится при попытке удалить объект ссылки внешнего ключа?

• ограничить (удалить нельзя) - запрет удалять целевой кортеж;пока есть хотя бы один ссылающийся на него кортеж;;

• каскадировать - удалить целевой кортеж; и все ссылающиеся.

2. Что должно случиться при попытке модифицировать потенциальный ключ, на который ссылается внешний ключ?

• ограничить;

• каскадировать.

При задании схемы БД разработчик должен указать не только множество атрибутов, составляющих потенциальный и внешний ключи, но и определить поведение СУБД при вышеперечисленных ситуациях.

1. Целостность атрибута. Значение атрибута должно быть из соответствующего ему домена.

### 4. Переход от структурной диаграммы к реляционной модели.

Структурная диаграмма - неформальная модель данных, которая служит для описания предметной области, используется как первое приближение схемы **БД.**

Сущность - вещь, которая существует и может быть выделена из других вещей того же рода.

Группа сходных сущностей - множество сущностей.

Атрибут: каждая сущность имеет свойство, называемое атрибутом, с каждым атрибутом ассоциируется значение, которое берётся из домена атрибута (множества его значений).

Ключ - атрибут или множество атрибутов, которые однозначно определяют сущность на множестве значений сущностей.

Отношение связывает один или более множеств сущностей.

Пример:

|  |  |  |
| --- | --- | --- |
| Сущности | Отношение | Тип отношения |
| детали | состоят из | замыкающее множество сущностей само на себя |
| склады детали | хранить | бинарное |
| поставщики детали проекты | поставлять | триадное |

Бинарные отношения классифицируются

1. Один-к-одному - каждой сущности из первого множества сущностей соответствует ровно одна в другом. Пример: НАЧАЛЬНИКИ, ОТДЕЛЫ.

2. Один-ко-многим - каждой сущности из первого множества сущностей соответствует несколько в другом. Пример: ГРУППЫ, СТУДЕНТЫ -ПРИНАДЛЕЖАТЬ.

3. Многие-ко-многим - каждой сущности из первого множества сущностей соответствует несколько в другом, каждой сущности из второго множества сущностей соответствует несколько в первом. Пример: СТУДЕНТЫ, ПРЕДМЕТЫ - УЧИТЬ.

4. Отношение "является"("is а"), если множество сущностей является подмножеством другого множества сущностей. Пример: НАЧАЛЬНИКИ, СОТРУДНИКИ.

Структурные диаграммы используют для наглядного преставления всей информации, которую необходимо ввести в БД.

**Преобразование структурных диаграмм в реляционную модель**

Преобразование множеств сущностей: результат - отношение, атрибутами которого являются атрибуты множества сущностей.

Преобразование отношения "является"("18 а"): в отношение, соответствующее бо

лее конкретному множеству сущностей, включающему внешний ключ на более общее.

# Сотрудники

“is a”

# Программисты

№ паспорта язык

Основной язык

|  |  |  |
| --- | --- | --- |
| **Программисты** | | |
| **ID\_Программиста** | **№ паспорта** | **Основной язык программирования** |
|  |  |  |

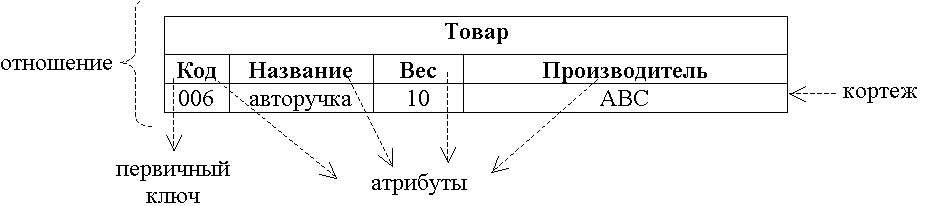
внешний ключ (FK)

Преобразование отношения один-к-одному: в оба отношения включают в качестве внешних ключей ссылки на другое отношение, один из ключей при этом можно убрать.

Преобразование отношения один-ко-многим: в отношении со стороны "многие" включают внешние ключи на отношение со стороны "один".

Преобразование отношения многие-ко-многим и парных отношений: создают вспомогательное отношение, которое содержит в себе ключи других отношений как внешние.

### 5. Реляционная алгебра. Операции реляционной алгебры.

Между структурной диаграммой и реляционной моделью имеется следующее соответствие:

Домен-множество значений, которые принимает атрибут.

Количество атрибутов - степень (арность) отношения.

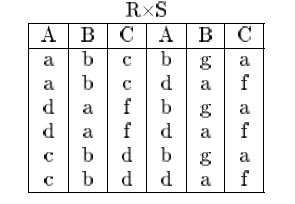
Первичный ключ - атрибут или совокупность атрибутов таких, что в любой момент времени не существует двух или более кортежей, содержащих одинаковые значения в этих атрибутах.

Переменная "отношение именованный объект, значение которого является отношением.

**Реляционная БД** - БД, воспринимаемая пользователем как набор отношений.

**Реляционная алгебра** состоит из операторов, использующих отношения в качестве операндов и возвращающих отношение в качестве результата. Операции делятся на группы:

традиционные операции над множествами **∪ ,∩, ⁄:**

пример:

RS

|  |  |  |
| --- | --- | --- |
| A | B | C |
| a  d  c | b  a  b | c  a  b |

|  |  |  |
| --- | --- | --- |
| A | B | C |
| b  d | g  a | a  f |

специальaные реляционные операции:

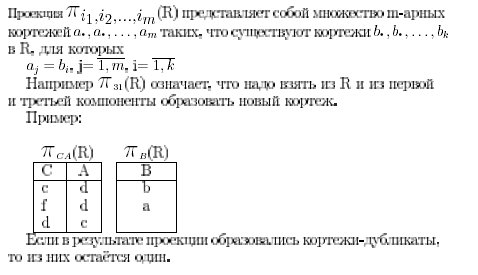
*выборка;*

*проекция;*

*соединение;*

*деление.*

**Проекция**

****

**Операция выборки**

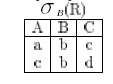
Пусть F - некоторая формула, которая включает в себя операнды, представляющих

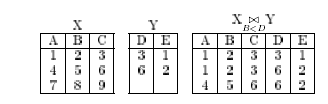
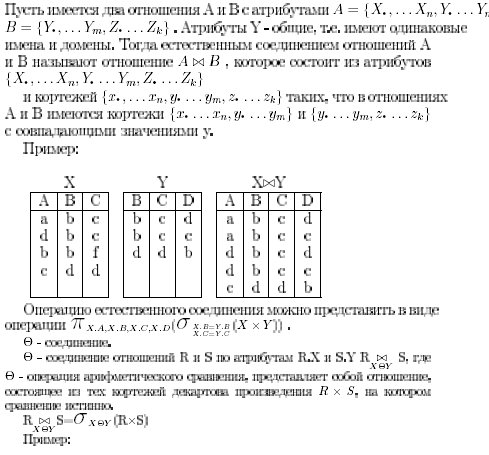
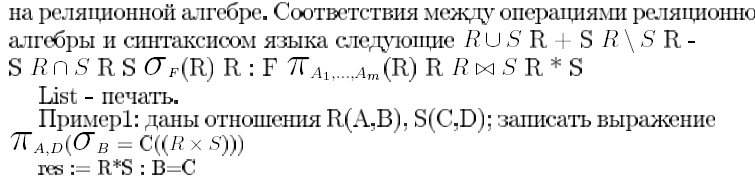
собой константы и имена атрибутов, а также операции сравнения и логические

операции

Выборка  из отношения R представляет собой отношение с теми же атрибутами, что и R, и со всеми такими кортежами, на которых формула F истинна.

Пример:

**Операция естественного соединения**

****

Покупатель (Имя, Адрес, Счёт)

Заказ (Номер, Имя, Товар, Количество)

Поставщик (Фирма, Адрес фирмы, Товар, Цена)

Эта **БД** поддерживается фирмой, которая поставляет товары покупателям на основе их заказов. Это фирма-посредник, которая покупает товары у поставщиков. Покупатели предоставляют фирме некоторое количество денег, которые отмечаютя в атрибуте Счёт.

Запрос!: напечатать всех должников

List Покупатель : Счёт < О

Запрос2: напечатать имена фирм, набор товаров и их цены для заказа покупателя с именем Петров.

List Заказ\*Поставщик : Имя = 'Петров'

ЗапросЗ: напечатать имена поставщиков, которые могут поставить все предметы, затребованные Петровым.

S := Поставщик I := Поставщик В:= Заказ : Имя = 'Петров' NS := 8\*1 - (Поставщик //фирма может поставить NSB := NS.(S\*B) //все пары Поставщик-Товар, где поставщик не может поставить //товары, заказанные Петровым. ListS - (NSB

### 6. Аномалии избыточности. Функциональные зависимости.

При проектировании реляционных БД следует выбрать такую ее схему , чтобы она была в некотором смысле лучшей. Рассмотрим вначале ,какими свойствами не должна обладать БД.

***Пример***: Задано отношение : Поставщики(Название , Телефон , Предмет ,Цена) отношение создает информацию о поставщиках товара. Для каждого поставщика имя и телефон ,для каждого товара название и цена.

|  |  |  |  |
| --- | --- | --- | --- |
| Название | телефон | предмет | цена |
| КИРС КИРС  РОСУАМ РОСУАМ | 235523 235523 330793 330793 | Compa Q 8086  Epson 8088  RV 286 RO  Intel 386 | 170 260 596 591 |

Ключ отношение название предмет. В этой схеме есть недостатки

1. В ней есть избыточная информация - дублируется номер телефона, последствия : увеличение файла БД , уменьшение эффективности запроса.

2. Аномалии изменения : бд потенциально может содержать противоречивую информацию - для одной и той же фирмы разные номера телефонов.

3. Аномалии вставки : если фирма ничего не продает , то нельзя сохранить информацию о фирме.

4. аномалия удаления: Если все предметы, то удалится информация о поставщике.

Аномалии - следствие избыточности.

Избыточность и все сопутствующие проблемы можно исключить , если отношение поставщики (Название , телефон , предмет , цена) разбить : на Фирмы(название, телефон ) и Товары (Название , Предмет , Цена)

Недостатки новой схемы : Если необходимо получить информацию и о товаре и о телефоне фирмы-продавца ,то необходимо сделать соединение 2-х таблиц. Операция соединения крайне трудоемка. В предметном варианте схемы эту операцию не нужно было проводить.

Основная проблема проектирования БД : Как избежать избыточности , как получить из плохой избыточности хорошую без избыточную.

Центральным понятием в нормализации БД (т. е. приведение к избыточному состоянию) является функциональные зависимости . Если 1 атрибут однозначно определяет значение другого , то говорят о функциональной зависимости. Простейший случай - ключ. Пример: Название фирмы функционально определяет телефон фирмы.

Определение: Пусть R - отношение . X,Y- произвольные подмножества множества атрибутов R. X функционально определяет Y. Y функционально зависит от X. Если для 2 кортежей с одинаковыми значениями атрибутов X , атрибуты Y будут также равны. *X*—>*Y*

Пример :Название —> Телефон

Пример :Название ,Предмет —> Цена

Единственный способ определить функциональную зависимость - анализ предметной области, функциональная зависимость - смысловое свойство атрибутов отношения, функциональные зависимости являются ограничениями целостности БД. При каждом обновлении БД все они должны быть проверены. В БД очень много функциональных зависимостей. Хотелось бы найти минимальное множество такое, что если все выполняется в БД , то выполняется и все оставшиеся функциональные зависимости.

### 7. Аксиомы Армстронга.

Центральным понятием в нормализации БД (т. е. приведение к избыточному состоянию) является функциональные зависимости. Если 1 атрибут однозначно определяет значение другого , то говорят о функциональной зависимости. Простейший случай - ключ. Пример: Название фирмы функционально определяет телефон фирмы.

Определение: Пусть R - отношение . X,Y- произвольные подмножества множества атрибутов R. X функционально опредеяет Y. Y функционально зависит от X. Если для 2 кортежей с одинаковыми значениями атрибутов X , атрибуты Y будут также равны. *X*—>*Y*

Пример :Название —> Телефон

Пример :Название,Предмет —> Цена

Если в отношении имеются некоторые функциональные зависимости , то из них можно вывести другие функциональные зависимости также выполняющиеся в этой БД.

Пример: Если в отношении имеется R некоторые функциональные зависимости , то из них можно вывести другие функциональные зависимости *А* —>*В* и *В* —>*С* , то в нем имеются функциональные зависимость *А* —>*С*

О**пределение:** пусть в R имеется 2 кортежа такие что А- равны , а С - нет. В соответствии с определением функциональной зависимости для этих 2 кортежей значение атрибутов В будут равны, т. к. имеет место функциональная зависимость *А* —>*В .* Из наличия функциональной зависимости *В* —>*С* следует , что значения атрибутов С равны. Противоречие доказывает утверждение .

**Определение:** Множество всех функциональных зависимостей ,которые выводятся из множества функциональных зависимостей F называется замыканием F и обозначается *F+*

Пример : Пусть задано отношение *R = ABC*и есть функциональная зависимость : *F = {А* —>*В, В* —>*С}*

*F+*состоит из следующих функциональных зависимостей вида *X*—>*Y:*

1. *А ∈X : ABC -> АС, АВ -> ВС, А* ->*А,...*

1. *А ∉XB∈X*

*A∉Y :BC ->B,B->C,B->0*

3. *С -> С,С -> О*

Задание правила вывода одних функциональных зависимостей из других.

Аксиомы:

1. рефлексивность : если , то *А* —>*В*

*2.* дополнение : если *А* —>*В* ,то *АС* —>*ВС*

3. транзитивность : если *А* —>B,В —> С, то A -> *С*

Аксиомы Армстронга непротиворечивы следовательно функциональные зависимости , которые получают с их помощью действительно содержатся в *F+.* Аксиомы Армстронга полны. Это значит что по этим правилам можно получить все функциональные зависимости которые есть в *F+.* Эти 2 утверждения позволяют утверждать , что вывод новых функциональных зависимостей на основе определения функциональной зависимости и при помощи Аксиом Армстронга эквивалентны.

Из Аксиом Армстронга можно получить произвольные правила:

Если *X -> У* и *X ->Z,* то *X ->YZ*



### 8. Замыкание множества атрибутов.

Цель :Найти множество неприводимых зависимостей.

Рассмотрим следующую задачу: Дано множество функциональных зависимостей F и функциональная зависимость *X —>Y.*

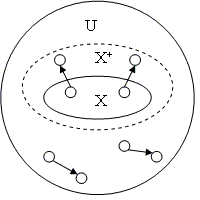
Необходимо определить : принадлежит ли *X*—>*Y*к замыканию *F+.* Иначе говоря выводима ли *X*—> У из множества F.

*F ->F+*

*X ->Y*

Для решения этой задачи используется замыкание атрибутов.

**Определение :** Пусть F - множество функциональных зависимостей , определенных на множестве атрибутов U (все атрибуты , которые могут встретится на F) и X подмножество U . Тогда оболочки X по отношению к F (обозначение *X*+) есть множество атрибутов А , таких что функциональные зависимости *X*—>*А* могут быть выведены из F при помощи Аксиом Армстронга.



**Алгоритм построения замыкания атрибутов**

Алгоритм приводится на псевдо коде.

Замыкание= *X*

hile (1)

для каждого *А* —>*В ∈F*

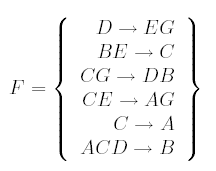
if*(А* ⊆Замыкание)

Замыкание = Замыкание ∪B;

if( замыкание не применялось за последнюю итерацию)

Break;

Пример:



Построим оболочку множества атрибутов CG. Начиная с CG пробегаем все множествао и выбираем два.



CG - является потенциальным ключом отношения . Атрибуты CG однозначно определяют все атрибуты отношения .

Исходя из вышесказанного следует , если *Х+ = U*то потенциальный ключ отношения содержится в X, чтобы его выделить поступаем так:

1. Из X удаляем произвольный атрибут *Х1 = Х\А*

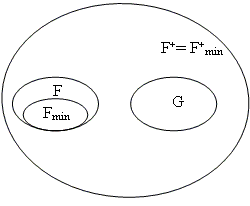
2. Если вновь * = U*, то атрибут А лишний и процедуру повторяем.

3. Продолжаем до тех пор пока не получим такое множество из которого нельзя выбросить ни одного атрибута .Это и будет потенциальный ключ.

Теперь решаем исходную задачу при помощи введенных операций . Необходимо определить истинно ли условие : *X*—>*Y*∈*F+*Оно будет верным , если будет выполнятся следующее условие *:Y*⊆*Х+.*

### 9. Неприводимое множество функциональных зависимостей.

Решим задачу : Найти минимальное множество функциональных зависимостей , из которых можно получить все остальные функциональные зависимости .рисунок



Пусть F1 и *F2-* два множества функциональных зависимостей . Если **то говорят что множество *F2*покрывает множество **Иначе говоря , любая функциональная зависимость ,выводимая из **может быть выведена и из *.*

Если **покрывает **и наоборот, то говорят что множества **и **эквивалентны.



Множество функциональных зависимостей является неприводимым , если выполняются свойства :

1. Часть каждой функциональной зависимости создает только 1 атрибут.

2. Левая часть любой функциональной зависимости является неприводимой . Это значит что если любой атрибут левой части опускается , то замыкание *F+*изменяется.

3.Ни одна функциональная зависимость в F не может быть выброшена из F без изменения *F+.*

Алгоритм построения неприводимого множества функциональных зависимостей.

1. Переписать функциональную зависимость т.о. , чтобы каждой из них в правой части имела единственный атрибут.

*А->В⇒А-> В,А->С*

То , что такое преобразование допустимо доказывается при помощи аксиом Армстронга.

2. Проверить можно ли опустить какой - либо атрибут из левой части функциональной зависимости.

Способ: Пусть имеется функциональная зависимость *:Х* — >*А, В* ∈*X*

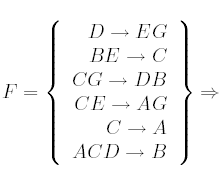
*А* ∈*(Х\В)+*

*(Х\В)* — >*А* ∈*F+*и В можно опустить из левой части функциональной зависимости .

3. Проверить можно ли опустить какую-либо функциональную зависимость.

Способ: Пусть имеется функциональная зависимость *X*— >*А.* Построим *Х+* над *{F\X*->*А}.* Если *А* ∈*Х+* , то *X -> А* ∈*{F\X ->A}+*и эту функциональную зависимость можно исключить .

Пример: Пусть имеется множество :

шаг2 : Функциональная зависимость СЕ , функционально определяет



А, исключается , т.к. Е м.б. опущен

 включает атрибут BCG→B можно исключить



Шаг 3:



### 10. Алгоритм поиска потенциальных ключей.

Цель :Найти множество неприводимых зависимостей.

Рассмотрим следующую задачу: Дано множество функциональных зависимостей F и функциональная зависимость *X —>Y.*

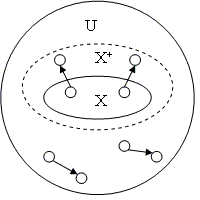
Необходимо определить : принадлежит ли *X*—>*Y*к замыканию *F+.* Иначе говоря выводима ли *X*—> У из множества F.

*F ->F+*

*X ->Y*

Для решения этой задачи используется замыкание атрибутов.

**Определение :** Пусть F - множество функциональных зависимостей , определенных на множестве атрибутов U (все атрибуты , которые могут встретится на F) и X подмножество U . Тогда оболочки X по отношению к F (обозначение *X*+) есть множество атрибутов А , таких что функциональные зависимости *X*—>*А* могут быть выведены из F при помощи Аксиом Армстронга.



**Алгоритм построения замыкания атрибутов**

Алгоритм приводится на псевдо коде.

Замыкание= *X*

hile (1)

для каждого *А* —>*В ∈F*

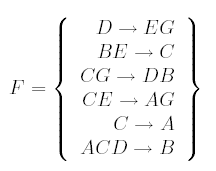
if*(А* ⊆Замыкание)

Замыкание = Замыкание ∪B;

if( замыкание не применялось за последнюю итерацию)

Break;

Пример:



Построим оболочку множества атрибутов CG. Начиная с CG пробегаем все множествао и выбираем два.



CG - является потенциальным ключом отношения . Атрибуты CG однозначно определяют все атрибуты отношения .

Исходя из вышесказанного следует , если *Х+ = U*то потенциальный ключ отношения содержится в X, чтобы его выделить поступаем так:

1. Из X удаляем произвольный атрибут *Х1 = Х\А*

2. Если вновь * = U*, то атрибут А лишний и процедуру повторяем.

3. Продолжаем до тех пор пока не получим такое множество из которого нельзя выбросить ни одного атрибута .Это и будет потенциальный ключ.

Теперь решаем исходную задачу при помощи введенных операций . Необходимо определить истинно ли условие : *X*—>*Y*∈*F+*Оно будет верным , если будет выполнятся следующее условие *:Y*⊆*Х+.*

### 11. Декомпозиция реляционных схем. Условия восстановимой декомпозиции.

Декомпозиция реляционной схемы *R = {A1,A2,...,An}* есть множество схем *р = {Ri, R2, . . . , Rk}* таких, что *.*

Цель декомпозиции: устранение избыточности, которая приводит к аномалиям.

Пример:

Пусть есть реляционная схема Поставщики (Название , телефон, предмет, цена)=НТПЦ. В схеме имеются следующие функциональный зависимости: *F =* {Н — > Т, НП — > Ц}. Чтобы устранить избыточность, НТПЦ-схема разделяется на две реляционные схемы:

, R1=НТ, R2=НПЦ

Пусть имеется отношение реляционной схемы *R =* НТПЦ. После декомпозиции получаем проекции **r** в схемы НТ и НПЦ: , Можно ли сказать, что r нт и r нпц содержат такую же информацию, что и **r**? Как восстановить обратно?

Чтобы получить исходное отношение г из проекций, следует выполнить обратную операцию соединения.



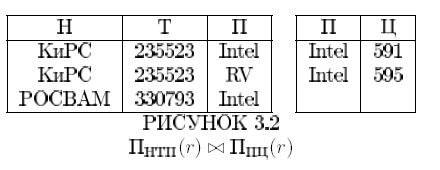
Необходимо ответить на следующий вопрос: при каких условиях **S=r**, - т.е., когда отношение восстановимо из своих проекций?

Пример:

Восстановимая декомпозиция

*R =* {НТПЦ}

|  |  |  |  |
| --- | --- | --- | --- |
| н | Т | П | Ц |
| КиРС КиРС РОСВАМ | 235523 235523 330793 | Intel RV  Intel | 591 596 595 |



|  |  |  |  |
| --- | --- | --- | --- |
| Н | Т | П | Ц |
| КиРС | 235523 | Intel | 591 |
| КиРС | 235523 | Intel | 595 |
| КиРС | 235523 | RV | 596 |
| РОСВАМ | 330793 | Intel | 591 |
| РОСВАМ | 330793 | Intel | 595 |

В общем случае, что подтверждает данный пример, соединение проекций не равно исходному отношению,



Чтобы получить восстановимую декомпозицию, т.е. когда отношение равно соединению его проекций, следует учитывать его функциональные зависимости. Верна следующая теорема:

Пусть *р ={Ri,R2} -* декомпозиция схемы R, F - множество функциональных зависимостей в R, тогда *р* имеет восстановимое соединение по отношению к F тогда и только тогда, когда  принадлежит *F+*

Пример:

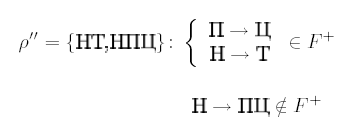
*R =* {НТПЦ} *F =* {Н -> Т, НП -> Ц}

1.

*р' =* {НТП,НПЦ}:

Используя ранее рассмотренные методы можно доказать, что рассмотренные ранее зависимости не принадлежат оболочке и данная декомпозиция не будет восстановимой.

2.



### 12. Многозначные функциональные зависимости.

**Определение функциональной зависимости**: Пусть R - отношение . X,Y- произвольные подмножества множества атрибутов R. X функционально определяет Y. Y функционально зависит от X. Если для 2 кортежей с одинаковыми значениями атрибутов X , атрибуты Y будут также равны. *X*—>*Y*

В базе данных может существовать избыточность, для описания которой не достаточно функциональных зависимостей.

Многозначная зависимость является обобщение функциональной и позволяет описать новый тип избыточности. Благодаря декомпозиции получается четвертая нормальная форма.

Если *X*— >*Y -* функциональная зависимость.

**Определение многозначной зависимости**: Пусть задана реляционная схема R и X,Y - множества атрибутов R. X многозначно определяет Y, или Y многозначно зависит от X, *X (X*— > У), если для любых двух кортежей S и Т таких, что *S[X] = Т[Х]* в отношении г будут существовать два кортежа U и V таких, что 1) *U[X] = V[X] = S[X] = Т[Х}-*

2) *U[Y] = T[Y];* U[R *\X\Y] = S[R\X\Y];*

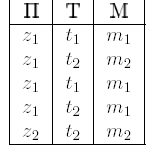
*3)* V[Y]=S[Y] ,V[R *\X\Y] = T[R\X\Y].*

Иначе говоря, если в отношении г имеются кортежи и то в отношении г будут иметься также следующие два кортежа: , .

Пример

Сбыт (Предприятия, Товары, Магазины)=С(П,Т,М);

Будем считать, что если предприятие выпускает какой-либо товар, то оно отправляет его во все магазины, с которыми связано.



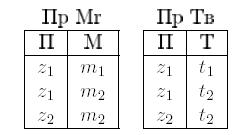
В этой схеме имеется многозначная зависимость П→>Т П→>М . Рассмотрим аномалии, присутствующие в данной схеме:

1. Аномалия вставки: если в таблицу следует занести вставку, что товар изменил название - следует изменить столько кортежей, со сколькими магазинами связано данное предприятие.

2. Аномалия удаления: если из таблицы следует отразить факт, что предприятие перестало выпускать какой-либо товар, то следует удалить столько кортежей, со сколькими магазинами связано данное предприятие.

Аномалии можно устранить следующей декомпозицией:

Из примера видно, что избыточность имеется.



### 13. Трехуровневая архитектура и SQL. Операторы CreateSchema и CreateView.

Принцип отделения логического уровня представления от физического лёг в основу обобщённой **СБД,** Эта структура получила название трёхуровневая архитектура: в ней имеется три уровня абстракций, на которых рассматривается **СБД:**

1. Концептуальный - описывает логическую структуру всей **БД.**

2. Внутренний - связан со способом хранения информации на физических устройствах хранения.

3. Внешний - связан со способом представления данных для пользователей.

. SQL- комбинация исчисления кортежей и реляционной алгебры.

***В SQL база данных -* это множество схем**. Каждая схема состоит из множества определений таблиц, представлений и прав доступа. Схема определяется следующим образом:

**CreateSchemaAuthorizationuser [элемент\_ схемы {, элемент\_ схемы}];**

элемент\_ схемы: =определение\_ таблицы | определение\_ вида | права\_ доступа

Создание таблиц осуществляется командой:

CreateTable имя\_ таблицы (определение\_ столбца {, определение\_ столбца});

определение\_ столбца:= имя\_ столбца тип\_ данных [NOTNULL];

тип\_ данных:= СНАRАСТЕR[(длина)] | NUMERIC[(precision[, scale])] | INTEGER | SMALLINT | FLOAT | REAL | DOUBLE PRECISION;

***Пример****:* Создать таблицу сотрудников, для которых хранится следующая информация: номер паспорта, год рождения, заработная плата.

Create Table Сотрудники(

Паспорт Character(15),

Фамилия Character(3O),

Имя Character(30),

ГодРождения Numeric(4),

должность Character(40), зарплата Numeric(lO));

Уровню внешнего представления в языке SQL соответствуют логические таблицы (виды), которые задаются командой CreateView, которая имеет вид:

**CreateView имя\_ вида[(определение\_ столбца{, определение\_ столбца})]**

**As select\_ предложение [With Check Option];**

Внешний уровень позволяет взять реально существующие таблицы и выбрать те данные на которые имеет права пользователь.

***Пример****:* Необходимо для пользователя создать вид, в котором из таблицы сотрудники удалены должность и заработная плата и в котором видны только те сотрудники, которые родились раньше 1975 года.

Create View сотрудники2 (Паспорт Character(15), Фамилия Character(30),

Имя Character(3O), Отчество Character(3O), Год\_ Рождения Numeric(4))

AsSelect Паспорт, Фамилия, Имя, Отчество, ГодРождения From сотрудники

Where ГодРождения<1975;

WithCheckOption - если указаны эти ключевые слова, то при дополнении и модификации View будет проверятся принадлежит ли новая запись виду.

*Пример:* При добавлении сотрудника с годом рождения 1980 запись не будет добавляться. На модификацию вида есть определенные ограничения.

*Модификации вида возможны если:*

1. Колонки вида - это колонки из других таблиц (нет вычислений над значениями из других таблиц).

2. В From части упоминаются таблицы, которые можно изменять.

3. Where часть не содержит подзапросов.

4. НетGroupByиHavingчастей.

### 14. SQL .Реализация ссылочной целостности внешних ключей.

Правила целостности можно задать в SQL двумя способами:

1. Декларативно

2. Процедурально

1) Декларативно задается при объявлении таблиц. В предложении CreateTable. В языке SQL объявляется в виде ограничений. Ограничения могут задаваться для всей таблицы или для столбца. Ограничения имеют синтаксис:

[Constraint имя\_ ограничения] ограничение

*Ограничения бывают:*

1. ***Unique*** *-* определение ключа-кандидата. Выделенный в ограничение столбец либо группа, должны содержать значения встречающиеся в таблице единственный раз. Если ключ-кандидат охватывает несколько столбцов, то его необходимо пометить в конце таблицы.

Для таблицы :

CREATETABLE АУДИТОРИИ( КОРПУС CHAR(2), НОМЕР CHAR(4), ТИП CHAR(5), UNIQUE (КОРПУС, НОМЕР) )

**Для столбца** CREATETABLE СОТРУДНИКИ(НомерПаспорта CHAR(20) UNIQUE, ФАМИЛИЯ CHAR(30),)

**2) *PrimaryKey****.* В таблице может быть единственный первичный ключ. Задается в двух формах: для таблицы, для столбца.

***3) ForeignKey*** *-* объявление внешнего ключа. Объявляет группу столбцов таблицы внешним ключом, который ссылается на другую таблицу. Допустима только форма задания для всей таблицы (для столбца отсутствует).

ForeignKey (перечень\_ столбцов) Reference имя\_ таблицы

[(перечень\_ столбцов)] ссылочная\_ спецификация;

Перечень столбцов можно не указывать если имена совпадают в первой и второй таблице. Ссылочная спецификация задает, что будет происходить при изменении (OnUpdate) или при удалении (OnDelete) родительского ключа. Возможные опции:

Cascade - распространяет удаление на все подчиненные. Если изменяется, то изменяются и все подчиненные.

SetNull - установить пустое значение в подчиненных записях.

SetDefault - установить значения по умолчанию в подчиненных записях.

**4) *Check*** *-* предикат проверяет выполнение условия. Можно устанавливать для столбца и для таблицы.

*Пример:* Create Table A(X Integer Check(X>=0), Y Integer Check(y>=0), Constraint MyCheck Check(x+y>0));

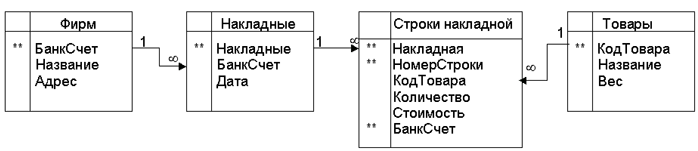
Первые две конструкции Check - для столбцов, третья - для таблицы. ForeignKey - отношение один ко многим. А как создать отношение один к одному?

Для этого:

1. В одной из таблиц группа полей объявляется внешним ключом (For­eignKey).

2. Потом объявляется Unique.

*Пример:* Пусть имеются следующие таблицы.



Описание таблиц на SQL имеет вид:

Create Table Фирмы(БанкСчет Char(12) Primary Key, Название Char(20), Адрес Char(40));

Create Table Товары(КодТовара Char(15) Primary Key, Название Char(30), Bee Decimal(lO));

CreateTableНакладные (НакладныеChar(20), БанкСчетChar(12), ДатаChar(lO), PrimaryКеу(Накладные,БанкСчет), ForeignKey (БанкСчет), Ref­erencesФирмыOnUpdateCascadeOnDeleteCascade);

CreateTableСтрокиНакладной(БанкСчетChar(12), НакладнаяChar(20), НомерСтрокиDecimal(3), КодТовараChar(15), КоличествоDecimal(lO), СтоимостьDecimal(lO), PrimaryКеу(БанкСчет,Накладные, НомерСтроки), ForeignKey (КодТовара), ReferencesТовары, ForeignКеу(БанкСчет,Накладные), ReferencesНакладные);

### 15. SQL .Реализация Соединения таблиц в операторе SELECT.

**SELECT оператор**

С его помощью можно выбрать данные из СУБД. Данные можно выбрать из нескольких таблиц. Оператор SELECT имеет следующий вид:

Select [Distinct] список \_ столбцов From источники

Where предикат GroupBy список\_ столбцов

Having предикат Order By {Столбец [Asc|Desc]};

*Результат запроса -* таблица, столбцы которой заданы в части Select.

1. Часть *From*определяет одну либо несколько таблиц, из которых извлекаются данные.

2. В части *Where*определяются критерии, которым должны удовлетворять строки, появляющиеся в итоговой таблице.

3. Часть *GroupBy*группирует выходные данные по одинаковым значениям перечисленных столбцов. Часть GroupBy применяется для выполнения агрегатных функций над группами строк.

4. Часть *Having*определяет критерии, которым должны удовлетворять группы строк, формируемые предложением GroupBy.

5. С помощью *OrderBy*можно расположить результаты запроса в определенном порядке.

6. Если задано ключевое слово *Distinct,* то если в результате выполнения запроса появляются одинаковые строки, то дубликаты убираются

*Пример 1:* Выбрать все товары вес которых больше 1 кг.

Select \* From Товары Where вес>1;

или

Select КодТовара, Название From Товары Where вес>1;

В первом случае \* значит выбрать все столбцы таблицы. Во втором случае указаны конкретные имена столбцов.

*Пример 2:* Выбрать названия товаров, код накладной, номер в строке для всех строк накладной в которых более 1000 единиц товаров.

Select СтрокиНакладной.Накладная, СтрокиНакладной.НомерСтроки,

Товары.Название From СтрокиНакладной, Товары

Where СтрокиНакладной.КодТовара=Товары.КодТовара

And СтрокиНакладной.Количество>1000;

*Альтернативный синтаксис* для соединения - использование ключевого слова Join в части From.

TablA [Natural] [тип\_ соединения] Join TablB.

В обеих таблицах выбираются столбцы с одинаковыми именами. В итоге помещаются только те строки из декартового соединения в которых столбцы с одинаковыми именами имеют одинаковые значения.

*Типы соединения:*

*• Inner -* соединяются только те строки, где найдены совпадения значений (используется по умолчанию).

• *Left -* результат все строки из tablA + строки из tablB, для которых значения совпадают.

• *Right -* все строки из tablB + строки из tablA с совпадающими значениями.

3. Соединение посредством предиката.

TabА [тип\_ соединения] ТаbВ On предикат

В соединение попадают только те пары строк, которые удовлетворяют предикату. Вернемся к примеру 2:

*Пример:*

Select Накладная, НомерСтроки, Название

From СтрокиНакладной Join Товары Where Количество>1000;

*Пример 1:* Вычислить вес товаров, перечисленных в строках накладной (для каждой).

Select Накладная, НомерСтроки, Название, Количество\*Вес As

Вес\_ В\_ Строке From СтрокиНакладной Join Товары;

*Пример 2:* Напечатать все номера накладных в которых упоминается "сахар".

SelectDistinct Накладная From СтрокиНакладной Join Товары

Where Название="сахар";

### 16. SQL .Коррелированный и не коррелированный подзапрос.

*Пример :* Выдать цену сахара и его количество для каждой строки накладной.

Where Название="сахар";

или

Select Количество, Стоимость From СтрокиНакладной

Where КодТовара= (Select КодТовара From Товары Where Название="сахар")

Это пример запроса с подзапросом. Сначала выполняется подзапрос, а затем основной запрос. Подзапрос должен при использовании его в операторах сравнения возвращать единственное значение в единственном столбце.

**В корелированном подзапросе есть ссылки на таблицу, которая имеется во внешнем запросе.**

*Пример 1:* Найти все фирмы которые сделали поставки 11.11.2002.

Вариант 1 (не коррелированный):

SelectdistinctНазваниеFromФирмыJoinНакладные

Wheredata=11.11.02;

Вариант 2 (корелированный):

SelectНазваниеFromФирмыWhere 11.11.02 In

(SelectДатаFromНакладныеWhereНакладные.БанкСчет=Фирмы.БанкСчет~

*Порядок выполнения запроса следующий:*

Сначала выбирается какаялибо фирма. Для нее выполняется подзапрос в котором находится множество дат, в которые фирма делала поставки. Затем проверяется условие (то что заданная 11.11.02 находится в этом множестве) и решается поступит ли фирма в окончательное множество или нет. Эта процедура выполняется для каждой фирмы. Коррелированные подзапросы сильно влияют на производительность, но иногда без них трудно обойтись.

*Пример 2:* Выдать названия фирм, которые ничего не поставили.

Select Название From Фирмы WhereNotExists ( Select \* From Накладные Where БанкСчет=Фирмы.БанкСчет);

Предикат Exists(NotExists) проверяет является ли множество, возвращаемое подзапросом не пустым (пустым).

*Пример 3:* Выбрать названия товаров у которых названия одинаковые, а коды разные.

Select Distinct Название From Товары Where Exists(

Select \* From Товары Товары2 Where Товары.Название=Товары2.Название

And Товары.Код<>Товары2.Код);

Новая синтаксическая конструкция - переименование (alias).

Если необходимо, например, соединить саму с собой таблицу, то возникает следующая проблема: необходимо синтаксически различать 2 различных ее появления. Для этого и используется переименование. В нашем примере товары в коррелированном подзапросе переименованы в Товары2. Теперь можно различать таблицу товары из внешнего запроса и подзапроса.

*Пример 4-'* Выдать количество поставок, которые фирмы сделали в сентябре.

Select Название, Count (Накладная) As КоличествоПоставок

From Фирмы Join Накладные Where Дата Between 01.09.02 And 31.09.02

GroupByБанкСчет;

Рассмотрим множество строк, которое возвращается оператором Select без части GroupBy. Все это множество разбивается на подмножества таким образом, что все записи с одинаковым банковским счетом попадают в одно подмножество. Над каждым подмножеством выполняется агрегирующая функция (для этого примера - количество элементов подмножества) результат которой и возвращается в запросе.

*Пример 5:* Расчитать вес поставки.

Select Накладная, Sum(Количество\*Вес) As ВесПоставки

From СтрокиНакладной Join Товары GroupBy Накладные;

*Пример 6:* Для каждой фирмы подсчитать число поставок вес которой превышает одну тонну.

Select Название, Count (Накладные) As КоличествоБолыпихПоставок

From Фирмы Join Накладные Where Накладная In (

Select Накладная From СтрокиНакладной Join Товары GroupBy Накладная

HavingSum (Количество\*Вес)> 1000)

GroupByБанкСчет;

В *Having*части записываются те условия, в которых используется агрегирующая функция.

### 17. SQL,Оператор Select. Функции подведения итогов. ?

С его помощью можно выбрать данные из СУБД. Данные можно выбрать из нескольких таблиц. Оператор SELECT имеет следующий вид:

Select [Distinct] список \_ столбцов From источники

Where предикат GroupBy список\_ столбцов

Having предикат Order By {Столбец [Asc|Desc]};

*Результат запроса -* таблица, столбцы которой заданы в части Select.

1. Часть *From*определяет одну либо несколько таблиц, из которых извлекаются данные.

2. В части *Where*определяются критерии, которым должны удовлетворять строки, появляющиеся в итоговой таблице.

3. Часть *GroupBy*группирует выходные данные по одинаковым значениям перечисленных столбцов. Часть GroupBy применяется для выполнения агрегатных функций над группами строк.

4. Часть *Having*определяет критерии, которым должны удовлетворять группы строк, формируемые предложением GroupBy.

5. С помощью *OrderBy*можно расположить результаты запроса в определенном порядке.

6. Если задано ключевое слово *Distinct,* то если в результате выполнения запроса появляются одинаковые строки, то дубликаты убираются

*Пример 1:* Выбрать все товары вес которых больше 1 кг.

Select \* From Товары Where вес>1;

или

Select КодТовара, Название From Товары Where вес>1;

В первом случае \* значит выбрать все столбцы таблицы. Во втором случае указаны конкретные имена столбцов.

EXISTS - это оператор, который производит верное или неверное значение, Это означает что он может работать автономно в предикате или в комбинации с другими выражениями. Он берет подзапрос как аргумент и оценивает его как верный если тот производит любой вывод или как неверный если тот не делает этого.

Этим он отличается от других операторов предиката, в которых он не может быть неизвестным.

NotExists Предыдущий пример дал понять что EXISTS может работать в комбинации с операторами Бул. Конечно, то что является самым простым способом для использования и вероятно наиболее часто используется с EXISTS - это оператор NOT.

Запросы могут производить обобщенное групповое значение полей точно также как и значение одного пол. Это делает с помощью агрегатных функций. Агрегатные функции производят одиночное

значение для всей группы таблицы. Имеется список этих функций:

COUNT -производит номера строк или не-NULL значения полей которые выбрал запрос.

SUM -производит арифметическую сумму всех выбранных значений данного пол.

AVG -производит усреднение всех выбранных значений данного пол.

MAX -производит наибольшее из всех выбранных значений данного пол.

MIN производит наименьшее из всех выбранных значений данного пол.

### 18. SQL,Оператор Select. Предложение Having.

**SELECT оператор**

С его помощью можно выбрать данные из СУБД. Данные можно выбрать из нескольких таблиц. Оператор SELECT имеет следующий вид:

Select [Distinct] список \_ столбцов From источники

Where предикат GroupBy список\_ столбцов

Having предикат Order By {Столбец [Asc|Desc]};

*Результат запроса -* таблица, столбцы которой заданы в части Select.

1. Часть *From*определяет одну либо несколько таблиц, из которых извлекаются данные.

2. В части *Where*определяются критерии, которым должны удовлетворять строки, появляющиеся в итоговой таблице.

3. Часть *GroupBy*группирует выходные данные по одинаковым значениям перечисленных столбцов. Часть GroupBy применяется для выполнения агрегатных функций над группами строк.

4. Часть *Having*определяет критерии, которым должны удовлетворять группы строк, формируемые предложением GroupBy.

5. С помощью *OrderBy*можно расположить результаты запроса в определенном порядке.

6. Если задано ключевое слово *Distinct,* то если в результате выполнения запроса появляются одинаковые строки, то дубликаты убираются

*Пример 1:* Выбрать все товары вес которых больше 1 кг.

Select \* From Товары Where вес>1;

или

Select КодТовара, Название From Товары Where вес>1;

В первом случае \* значит выбрать все столбцы таблицы. Во втором случае указаны конкретные имена столбцов.

*Пример 2:* Выбрать названия товаров, код накладной, номер в строке для всех строк накладной в которых более 1000 единиц товаров.

Select СтрокиНакладной.Накладная, СтрокиНакладной.НомерСтроки,

Товары.Название From СтрокиНакладной, Товары

Where СтрокиНакладной.КодТовара=Товары.КодТовара

And СтрокиНакладной.Количество>1000;

Having может использовать аргументы которые имеют одно значение на группу вывода

*Пример 6:* Для каждой фирмы подсчитать число поставок вес которой превышает одну тонну.

Select Название, Count (Накладные) As КоличествоБолыпихПоставок

From Фирмы Join Накладные Where Накладная In (

Select Накладная From СтрокиНакладной Join Товары GroupBy Накладная

HavingSum (Количество\*Вес)> 1000)

GroupByБанкСчет;

В *Having*части записываются те условия, в которых используется агрегирующая функция.

### 19.SQL, Операторы изменения данных.

Добавление записей осуществляется оператором *INSERT:*

*Синтаксис:*

InsertInto имя\_ таблицы[(имя\_ столбца,...)] запрос | конструктор значений

*Пример 1:* Форма с конструктором значений.

InsertInto Товары(Код,Название,Вес) Values(2367,"Сахар",!);

*Пример 2:* Форма с выражением запроса.

InsertInto Товары Select КодТов,Назв,Вес From СписокТоваров Where Назв Like "С\*";

Этот запрос копирует записи из одной таблицы в другую.

Оператор удаления записей *Delete*

*Синтаксис:*

DeleteFromимя\_ таблицы [Whereпредикат];

*Пример:* Delete From Накладные Where data<"05.05.1995";

*Оператор модификации данных*

*Синтаксис:*

Update имя\_ таблицы set имя\_ столбца=выражение,... Whereпредикат;

*Пример:* Update Товары SetHa3BaHHe="Caxap"Where Название="Сахар";

# Практическийраздел

ИНДИВИДУАЛЬНЫЕ ПРАКТИЧЕСКИЕ РАБОТЫ, ИХ ХАРАКТЕРИСТИКА

### Введение

По результатам работы студентом должен быть представлен и защищен отчет. Содержание отчета включает:

1. Введение, содержащее постановку задачи.
2. Распечатку программы.
3. Вывод.

### ИПР №1

#### Часть 1.1 SQL: WHERE, JOIN

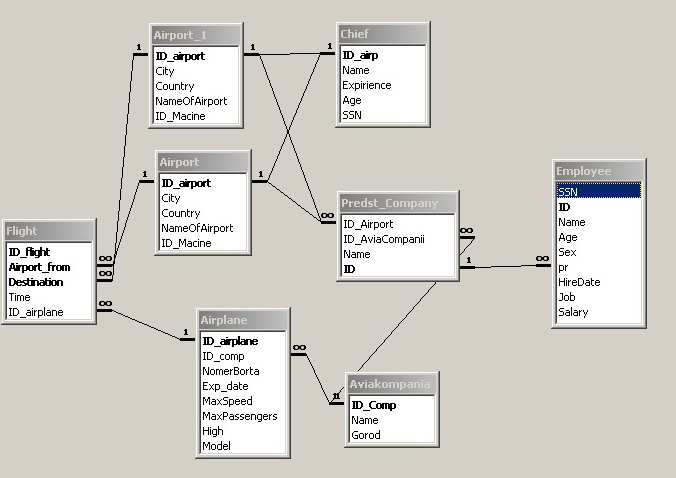
**Цель:** Получение практических навыков построения выборок с использованием секции WHERE оператора SELECT а также построения выборок из нескольких таблиц с использованием различных приемов объединения таблиц.

**Общая постановка задачи:** Построить выборки с использованием секции WHERE оператора SELECT. Построить выборки из нескольких таблиц с использованием различных приемов объединения таблиц.

**Практическая часть:**

**Вариант 1**

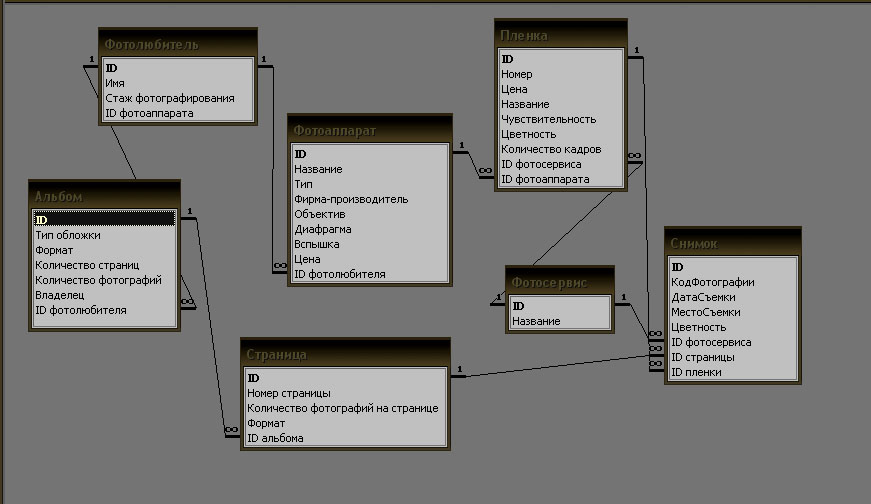
«Аэропорт»



1. Выбрать значения полей Name, Age и SSN таблицы Employee, где в номере телефона неизвестны 2 последние цифры - Phone = 55512??.
2. Выбрать значения полей Name, Age и Sex таблицы Employee, где возраст сотрудника превышает 30 лет.
3. Выбрать значения полей SSN, Name и Age таблицы Employee где в поле Address встречается слово «Masherova».
4. Выбрать значения полей Name, SSN, Phone таблицы Employee с условием, что сотрудник проживает по адресу «Gikalo,5» и возраст сотрудника может быть больше 30 лет, но меньше 50.
5. Вывести все возможные сочетания значений двух столбцов Predst\_Company.Name, Employee.Name таблицы Restaurant и Purveyor.
6. Вывести работников, по адресу которых проживает еще какой-либо работник аэропорта. Таким образом, по одному и тому же адресу может проживать 2 и более сотрудников.
7. Выбрать значения полей Name, Gorod для тех авиакомпаний, которые имеют одно и тоже имя с представительством авиакомпании.
8. Выбрать значения полей Name, Gorod для тех авиакомпаний, которые не имеют одно и тоже имя с представительством.
9. Для каждой авиакомпании вывести названия их (авиакомпании) представительств.
10. Для каждой авиакомпании выбрать все представительства, где в названии присутствует слово “Flights”.
11. Выполнить п.п. 6-8, 10 с использованием секции WHERE вместо JOIN.

**Вариант 2**

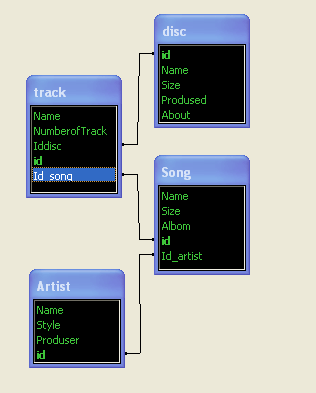
«Фотоальбом»

****

1. Выбрать значения полей ID, Format, NumberOfPages, NumberOfPhotos таблицы Album, где Cover = «Leather».
2. Выбрать значения полей ID, Format, Cover, NumberOfPhotos таблицы Album, где значение поля NumberOfPages между 60 и 200.
3. Выбрать значения полей ID, Format, NumberOfPhotos, NumberOfPages таблицы Album, где в значении поля Cover встречается сочетание «th». (Используйте оператор LIKE).
4. Выбрать значения полей ID, Format, Cover, NumberOfPhotos таблицы Album, где значение поля NumberOfPages больше 80 и Cover = «Leather».
5. Вывести все возможные сочетания значений двух столбцов Film.Name и Photograph.ShootingPlace таблиц Film и Photograph.
6. Найти в базе пленки, для которых в базе есть пленки с таким же названием.
7. Выбрать значения полей Format, Album\_ID для страниц из альбомов с пластиковой обложкой.
8. Выбрать значения полей Name, Producer, Price для тех фотоаппаратов, которые имеют название, отличное от названия своей фирмы-производителя.
9. Вывести все альбомы и соответствующие им страницы.
10. Для каждого пленки вывести снимки, у которых в названии места съемки присутствует слово «Экскурсия».
11. Выполнить п.п. 6-8, 10 с использованием секции WHERE вместо JOIN.

**Вариант 3**

«Музыкальные диски»

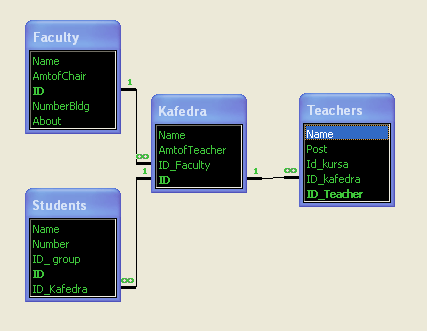
****

1. Выбрать название и размер диска Nightwish.
2. Выбрать название и размер всех дисков с размером меньше чем 600**.**
3. Выбрать название и размер диска в чьем названии содержится слово Day .
4. Выбрать название, производителя и описание всех дисков у которых размер больше 600 или производителем является фирма newCD**.**
5. Вывести все возможные сочетания значений двух столбцовпроизводителя и название трека**.**
6. Найти диски, которые были произведены одной и той же фирмой..
7. Выбрать название диска и название трека где Id диска и номер трека совпадает.
8. Для каждого диска выбрать название, цену и тип к которому относится диск (таблица **Costgrd**поле **Type**).
9. Для каждого диска выбрать название всех их треков.
10. Выбрать название диска и название трека для всех дисков у которых количество треков не превышает семи.

11. Выполнить п.п. 6-8, 10 с использованием секции WHERE вместо JOIN.

**Вариант 4**

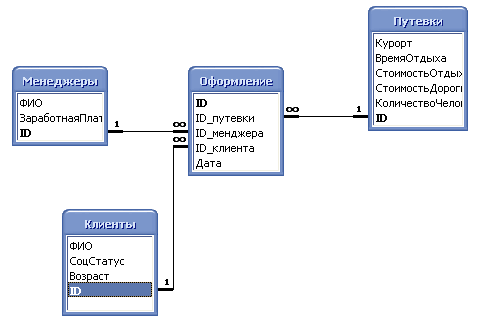
«Университет»

****

1. Выбрать название и адрес факультета КСиС.
2. Выбрать название и адрес всех факультетов с количеством кафедр больше трех**.**
3. Выбрать название и адрес всех факультетов находящиеся на улице «Бровки».
4. Выбрать название ,адрес и количество кафедр всех факультетов по адресу **«**Бровки 4» и с количеством кафедр больше трех**.**
5. Вывести все возможные сочетания значений двух столбцовадрес факультета и название кафедры**.**
6. Найти факультеты, которые располагаются по одному и тому же адресу.
7. Выбрать название факультета и название кафедры где количество кафедр и количество преподавателей совпадает.
8. Для каждого студента выбрать имя, размер стипендии и класс к которому относится стипендия (таблица **Stgrade** поле**Avgmark**).
9. Для каждого факультета выбрать название всех их кафедр.
10. Для каждого факультета выбрать все их кафедры где количество преподавателей меньше десяти.
11. Выполнить п.п. 6-8, 10 с использованием секции WHERE вместо JOIN.

**Вариант 5**

«Турфирма»

****

1. Выведите всех клиентов старше 20 лет, но моложе 35.
2. Создайте список клиентов, которых обслуживали менеджеры Розенкранц, и Горацио.
3. Выберите все курорты, в названии которых присутствует «остров».
4. Выведите всех клиентов, которых менеджер Козлова отправила в Альпы.
5. Выведите все возможные сочетания менеджеров и клиентов.
6. Сделайте список клиентов обслуживаемых одним менеджером.
7. Сделайте список менеджеров, отправлявших клиентов в Крым.
8. Выведите список курортов, на которые менеджер Карпович не отправлял клиентов.
9. Выведите все данные из таблицы Клиенты и все данные из соответствующей ей таблицы Менеджеры.
10. Выведите список курортов, на котором клиенты отдохнули благодаря работе менеджера Розенкранца.
11. Выведите даты оформления заказов для всех клиентов, не используя оператор JOIN.

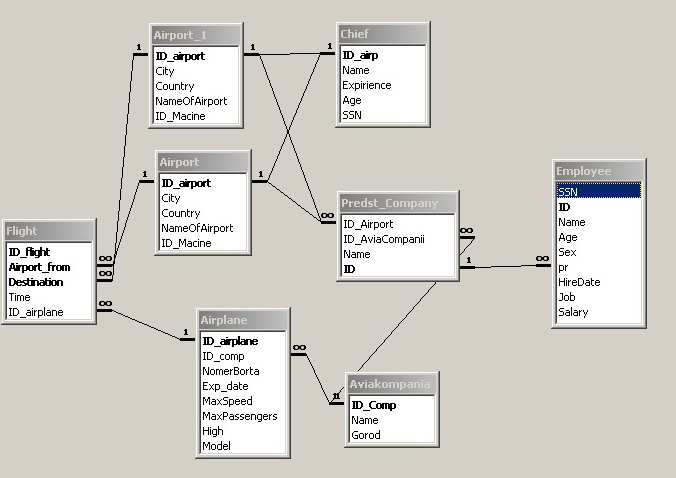
#### Часть 1.2 SQL\*Plus форматирование вывода, переменные подстановки

**Цель:**Получение практических навыков применения подстановочных переменных для получения данных от пользователя в процессе выполнения командных скриптов и реализации интерактивных запросов, а также изучение возможностей SQL\*Plus для форматирования вывода результатов запроса и построения простейших отчетов.

**Общая постановка задачи:** Получить данные от пользователя в процессе выполнения командных скриптов и реализации интерактивных запросов. Изучить возможности SQL\*Plus для форматирования вывода результатов запроса и построения простейших отчетов.

**Вариант 1**

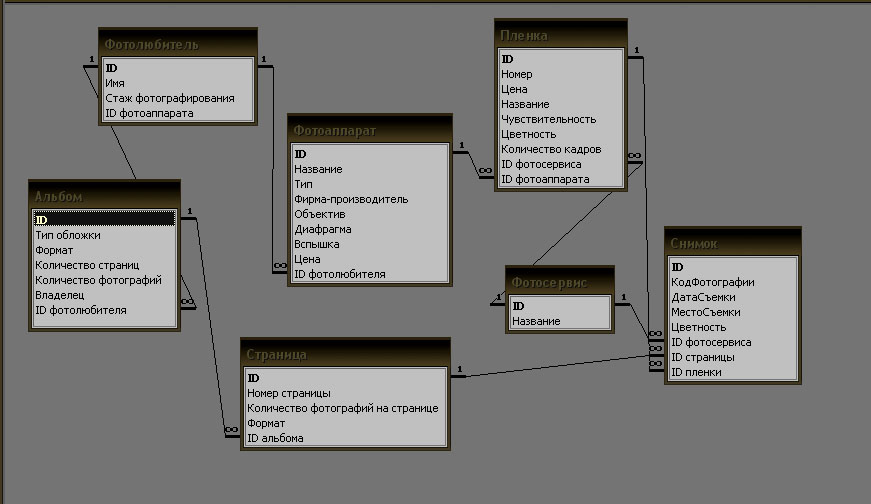
«Аэропорт»



1. Выбрать имя, пол, возраст сотрудника с условием, что возраст сотрудника составляет подстановочную переменную, задаваемую во время исполнения запроса.
2. Выбрать имя, пол, возраст сотрудника с заданным условием во время исполнения запроса – условия для выборки ввести через подстановочную переменную. Также задать порядок сортировки выводимых записей через подстановочную переменную.
3. Вывести список значений из таблицы Employee, запросив столбец, по которому необходимо выполнить сортировку, добавить этот столбец в список вывода. Запрашивать столбец только 1 раз.
4. Подготовить отчет по таблице Employee, запросив у пользователя критерий для выборки строк в отчет. В отчете вывести и указать псевдонимы Nick , Social, Male/Female, How\_old к столбцам соответственно: Name , SSN , Sex , Age. Также вывести верхний колонтитул – “EmployeeReport”, указать ширину записи для текстовых полей – 15 символов, ширина строки должна составлять 60 символов.

**Вариант 2**

«Фотоальбом»

****

1. Выбрать значения полей ID, Cover, Photographer\_ID для альбомов, формат которых равен некоторому значению (например, «10 х 15»). Значение запрашивается у пользователя во время выполнения запроса.
2. Выбрать строки из таблицы Album по некоторому условию. Условие для выборки ввести через подстановочную переменную. Также задать порядок сортировки выводимых записей через подстановочную переменную.
3. Вывести список значений из таблицы Album, запросив столбец, по которому необходимо выполнить сортировку, добавить этот столбец в список вывода последним. Запрашивать столбец только 1 раз.
4. Подготовить отчет по таблице Album, запросить у пользователя ID альбома.

Элементы отчёта:

а) Заголовок отчёта (верхний колонтитул) «Мой Альбом »– по левому краю, в одну строку.

б) Нижний колонтитул – «только для домашнего просмотра»

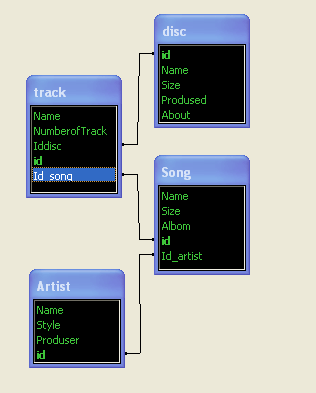
в) Столбцы NumberOfPages, NumberOfPhotos вывести под именами «Pages» и «Numberofphotos» соответственно, «Numberofphotos» - в две строки.

г) Остальные столбцы вывести под своими именами.

Ширина строки отчёта – 50 символов.

**Вариант 3**

«Музыкальные диски»

****

1. Выбрать название диска, где его размер меньше значения, которое вводится пользователем с консоли.

2. Выбрать название диск. Условие для выборки вводится с консоли пользователем.

Пример: Size=566**.**

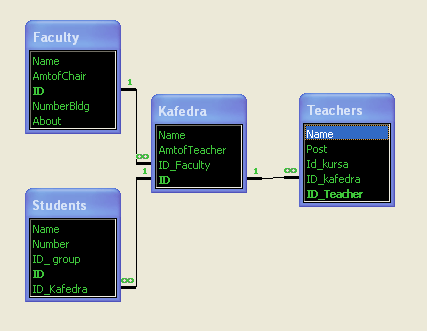
3. Выбрать название диска ,его описание(поле **About**) и поле, которое вводится с консоли пользователем. Произвести сортировку по полю вводимому с консоли.

4. Создать отчет по всем дискам включающий в себя название диска, размер и производитель. Выбрать только те диски где размер больше 500 но меньше 700.

В центре в заглавии вывести «Стандарт размера для дисков» а внизу по середине вывести «CD-shop». Переименовать столбцы **Name, Size**и **Prodused** в Название диска, Размер в Мбайтах и фирма-производитель.

**Вариант 4**

«Университет»

****

1. Выбрать название факультета где количество кафедр больше чем ColChair. ColChair вводится пользователем с консоли.

2. Выбрать название кафедры. Условие для выборки вводится с консоли пользователем.

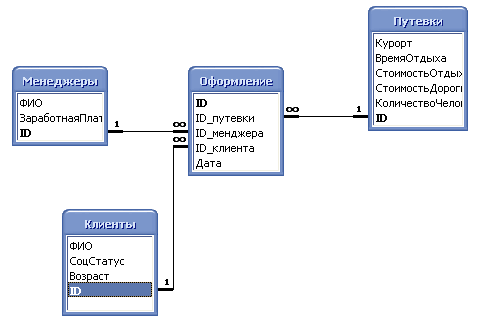
Пример: **Amtofchair>4.**

3. Выбрать название факультета ,его описание(поле **About**) и поле, которое вводится с консоли пользователем. Произвести сортировку по полю вводимому с консоли.

4. Создать отчет по всем факультетам включающий в себя название факультета, адрес и количество кафедр. Выбрать только те факультеты где количество кафедр больше четырех. В центре в заглавии вывести «Список факультетов» а внизу по середине вывести «Только для абитуриентов». Переименовать столбцы **Name, Amtofchair**и**NumberBldg**в Название факультета, Количество кафедр и адрес.

**Вариант 5**

«Турфирма»

****

1. Выведите имя менеджера, даты его заказов и имена клиентов, запросив имя менеджера с консоли.
2. Выведите информацию о путевках определенного клиента, запросив с консоли столбец для вывода и условие отбора (используйте две подстановочные переменные).
3. Выведите ФИО клиента, его возраст, и ещё одно поле, запрошенное с консоли, сортировку выполните именно по нему.
4. Составьте отчет, который будет запрашивать ФИО менеджера и выводить информацию о его заказах и клиентах в такой последовательности столбцов: «Имя клиента», «Возраст клиента», «Дата оформления заказа», «Название курорта», «Стоимость отдыха». В верхнем колонтитуле разместите имя менеджера, в нижнем – его заработную плату.

### ИПР №2

#### Часть 2.1 SQL: функции

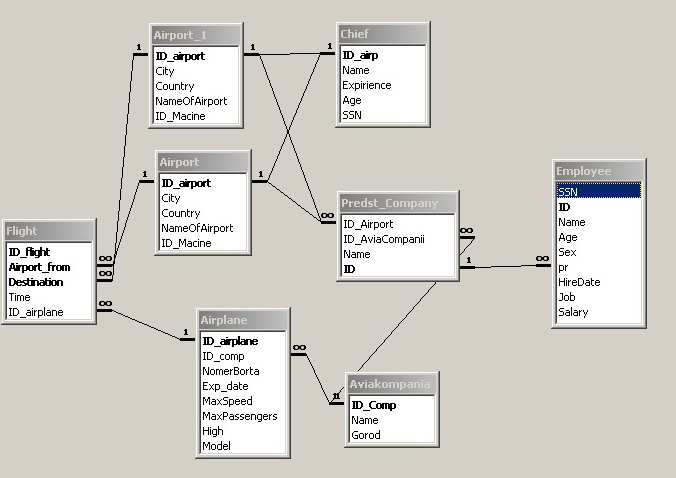
**Цель:**Изучение различных групп встроенных однострочных функций в SQL и получение навыков их использования.

**Общая постановка задачи:** Изучить различные группы встроенных однострочных функций в SQL и получить навыки их использования.

**Практическая часть:**

**Вариант 1**

«Аэропорт»



1. Выбрать имя, номер социального страхования , телефон из таблицы Employeec условием, что число букв имени сотрудника составляет 15 букв и начинается с буквы “A”.
2. Выбрать имя и номер социального страхования из таблицы Employee, в которых остаток от деления числа символов имени на число символов SSN равен 1.
3. Выбрать имя, телефон из таблицы Employee и отобразить количество дней со дня приема на работу сотрудника.
4. Выбрать имя, телефон из таблицы Employee и отобразить количество месяцев со дня приема на работу сотрудника.
5. Выбрать имя, фамилию, отобразить дату приема на работу в формате “[число] of [месяц] [год]” из таблицы Employee.
6. Выбрать имя, номер социального страхования из таблицы Employee. В случае, если номер социального страхования содержит пустое значение , заменить его на слово “Unknown”.
7. Выбрать имя, номер телефона сотрудника из таблицы Employee. При выборе значений из столбца Name заменить вхождение следующих имен:

|  |  |
| --- | --- |
| Исходный текст: | Заменить на текст: |
| Evgen | Yauheni |
| Alex | Alexandr |
| Slava | Vyacheslav |
| Dima | Dmitrii |

1. 8. Выбрать возраст и имя сотрудника из таблицы Employee. Возраст отобразить с помощью последовательности символов “\*”. 5 лет возраста сотрудника принять за один символ “\*”. Использовать функцию rpad.

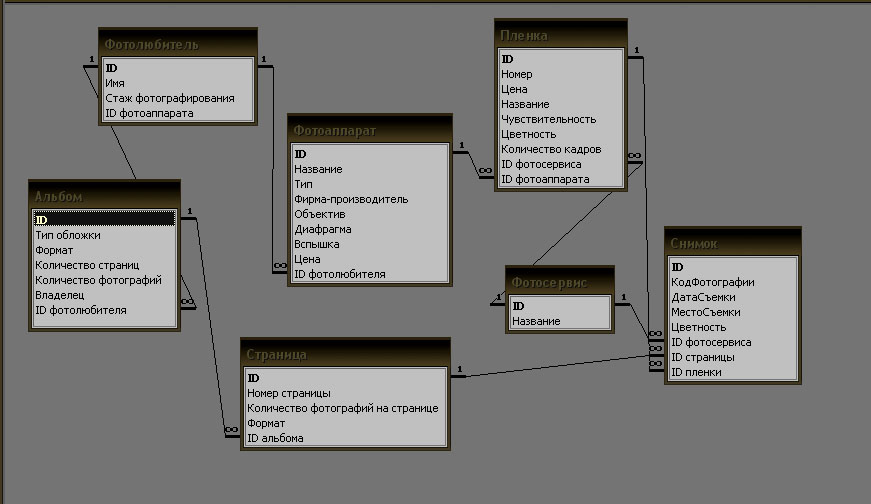
Например:

Ivanov \*\*\*\*\*\*

Petrov \*\*\*\*\*

**Вариант 2**

«Фотоальбом»

****

1. В таблице Photograph(Снимок) выбрать строки, где длина поля ShootingPlace больше 6 символов.
2. В таблице Album выбрать строки, подсчитав в отдельном столбце остаток отношения числа фотографий к числу страниц, для альбомов формата 9 х 13.
3. Выбрать строки таблицы Photograph, для каждой отобразить время в минутах, прошедшее с даты съемки до текущего момента.
4. Выбрать строки таблицы Photograph, для каждой отобразить время в неделях, прошедшее с даты съемки до текущего момента, округленное к ближайшему целому числу.
5. Выбрать строки таблицы Photograph, для каждой отобразить дату съемки в формате: «[число\_прописью] of [месяц\_прописью] [год]».
6. Отобразить поля ID, NumberOfPhotos таблицы Album, в случае пустых значений поля NumberOfPhotos выводить строку «альбом не заполнен».
7. Отобразить поля Name, Sensibility, Price таблицы Film, отображая значение поля Price следующим образом: если чувствительность(Sensibility) = 400 единиц, цена возрастает на 20%, если чувствительность(Sensibility) = 200 единиц, цена возрастает на 5%, цена для остальных значений Sensibility не изменяется.
8. По таблице Film выведите список, отображающий название плёнки, а также колонку VisualSensibility, отображающую чувствительность пленки в виде строки звёздочек. Каждая звёздочка соответствует ста единицам чувствительности. Сортируйте список по убыванию. При выполнении задания используется функция rpad

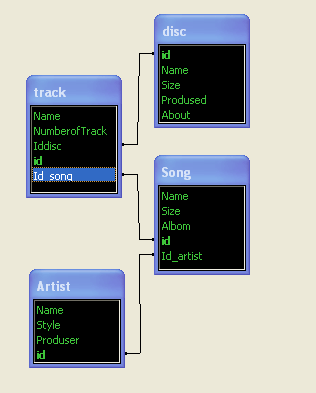
Например:

Konica \*\*\*\*

Kodak Gold \*\*

**Вариант 3**

«Музыкальные диски»

****

1. Выбрать название всех дисков где название написано неправильно(т.е. не первая буква в верхнем, а остальные в нижнем).
2. Выбрать название и остаток от деления размера диска на 230.

Пример: 654—>194.

1. Выбрать название диска и время, прошедшее с момента, прошедшего с даты создания.
2. Выбрать название диска и дату последнего дня месяца ,когда диск был создан.
3. Выбрать название и размер диска. Размер диска вывести в формате показанном в примере.

Пример: 235🡪 235,000.

1. Выбрать название диска и производителя. Строки поля **Prodused** со значением NULL вывести со значением «пиратская версия».
2. Выбрать название диска ,размер и цену. Для дисков у которых фирма-производитель:

newCD - Увеличить цену в 1.1 раз.

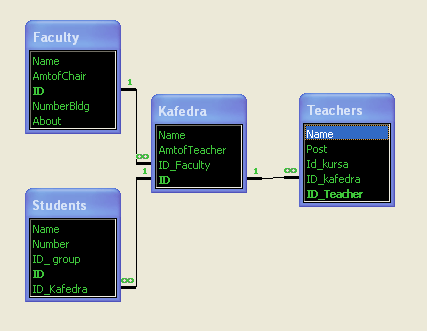
Звездная\_серия - Увеличить цену в 1.5 раз.

NULL – Уменьшить цену в два раза.

1. Сделать запрос который выводит название диска и его размер в виде некоторого количества символов «\*»(используя функцию rpad).

**Вариант 4**

«Университет»

****

1. Выбрать название факультета где встречаются буквы нижнего регистра.

Пример: КСиС.

1. Выбрать имя студента и его средний бал округленный до первого знака после запятой.

Пример: 4.33--->4.3.

1. Выбрать имя студента и время, прошедшее с момента, когда он поступил в учебное заведение, до текущего.
2. Выбрать имя студента и дату поступления студентов поступивших в первом наборе(которые поступали летом).
3. Выбрать имя студента и дату поступления. Дату студента вывести в формате показанном в примере.

Пример:Seventh of February 1981 08:00:00 AM.

1. Выбрать название факультета и описание(About). Строки поля About со значением NULL вывести со значением «неизвестно».
2. Выбрать имя студента ,средний бал и размер стипендии. Для студентов у которых средний бал:

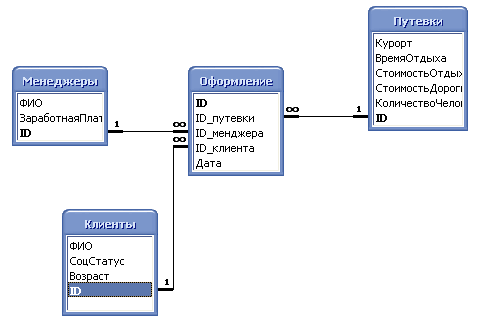
от 3 до 4 - Увеличить стипендию в 1.1 раз.

от 4 до 5 - Увеличить стипендию в 1.5 раз.

1. Сделать запрос который выводит имя студента и размер его стипендии в виде некоторого количества символов «\*»(используя функцию rpad).

**Вариант 5**

«Турфирма»

****

1. Выведите информацию о всех клиентах, выделив название курорта заглавными буквами.
2. Выведите имя менеджера и его зарплату, как количество полных сотен.
3. Выведите имена клиентов и названия курортов, для тех путевок, которые ещё не закончились.
4. Выведите названия курортов, дату оформления путевки и имена клиентов, указав дату и день недели их возвращения.
5. Выведите имена менеджеров, названия курортов и дату оформления заказа, отобразив число и месяц словами.
6. Выведите информацию о всех клиентах, заменяя пустое значение в поле «СоцСтатус» на фразу «нет информации».
7. Выведите имена клиентов, их возраст, имя менеджера, его зарплату, дату оформления путевки и название курорта, добавив к зарплате менеджеров у которых клиенты моложе 25 лет 10%.
8. Создайте диаграмму популярности курортов, в качестве индикаторов используйте символ «\*» (функция rpad).

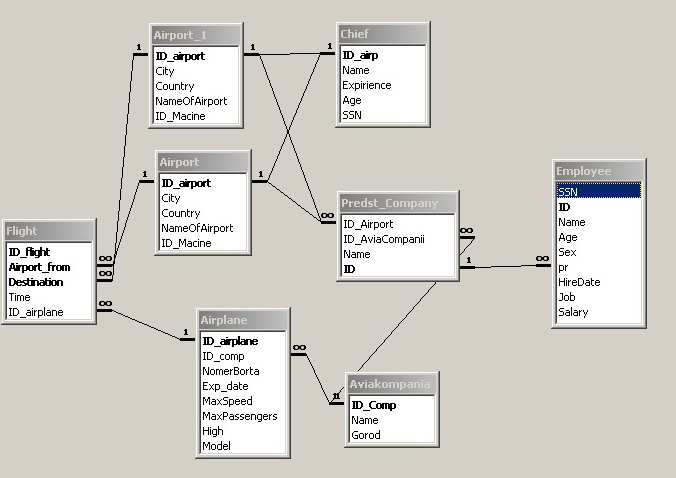
#### Часть 2.2 SQL: группировки

**Цель:** Изучение средств группировки данных выборки с помощью секции GROUPBY оператора SELECT и последующего ограничения выборки с помощью секции HAVING.

**Общая постановка задачи:** Группировать данные выборки с помощью секции GROUPBY оператора SELECT. Изучить средства последующего ограничения выборки с помощью секции HAVING.

**Вариант 1**

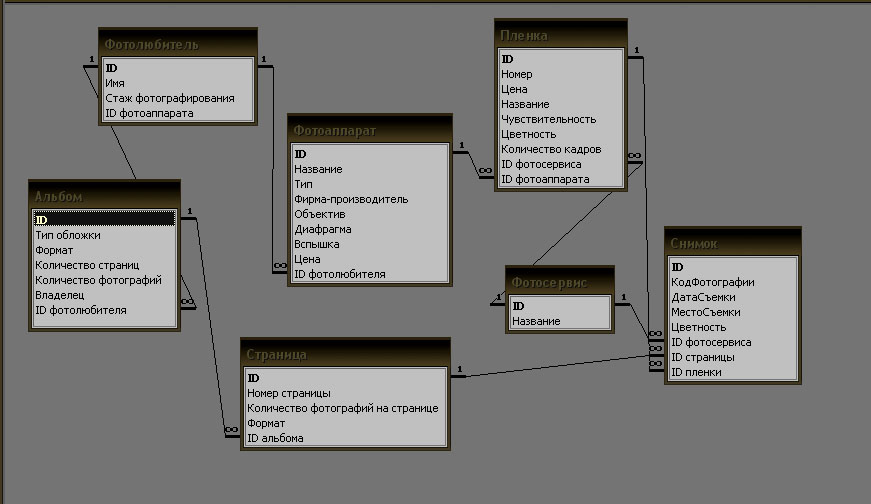
«Аэропорт»



1. Выбрать из таблицы Employee самого старого и самого молодого сотрудника , у которых есть SSN. Вывести также средний возраст сотрудников.
2. Отобразить число строк таблицы Employee, в столбце SSN присутствует непустое значение.
3. Отобразить число различных значений столбца Job.
4. Вывести средний возраст сотрудников, считая пустое значение как ноль.
5. Вывести название аэропорта и количество представительств для каждого аэропорта.
6. Посчитать количество аэропортов в каждом городе всех стран в базе.
7. Выделить группы по паданию возраста сотрудника в диапазон длиной в 10 лет, начиная с 20 лет и заканчивая 60 годами. Подсчитать количество сотрудников в каждой группе и вывести.
8. Вывести должности, где средняя заработная плата превышает 100у.е.
9. Вывести должности, где средняя заработная плата превышает 200у.е. и в должности присутствует слово “WAIT”.
10. Подсчитать среднюю зарплату по должностям; вывести минимальную заработную плату, сформированную из средних значений.

**Вариант 2**

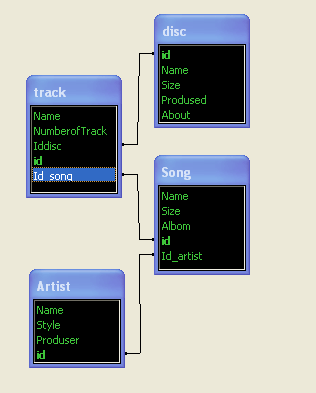
«Фотоальбом»

****

1. Отобразить минимальную и максимальную цену пленки, в названии которой есть слово «Kodak».
2. Отобразить общее число строк в таблице Film.
3. Отобразить число различных значений в столбце Sensibility таблицы Film.
4. Подсчитать среднее количество фотографий в альбомах, считая пустые значения поля NumberOfPhotos за 0.
5. В таблице Film выделить группы по равенству поля Sensibility и подсчитать в каждой группе среднюю цену пленки.
6. В таблице Film выделить группы по равенству поля Sensibility и в них подгруппы по равенству поля Price, и подсчитать максимальное количество кадров на пленке (NumberOfFrames) в подгруппах каждой группы.
7. В таблице Camera выделить группы по попаданию значения поля Price в некоторый диапазон: 0-99 – 1-я группа, 100-200 – 2-я группа… и подсчитать количество строк в каждой группе.
8. В таблице Film выделить группы по равенству поля NumberOfFrames (количество кадров). Отобразить только группы, в которых максимальная цена пленки меньше заданного значения (например, 60).
9. В таблице Film выделить группы по равенству поля Sensibility, отобрав только группы, в названии которых встречается «Konica». Подсчитать среднее количество кадров на пленке (NumberOfFrames) в группе. Отобразить только группы, в которых среднее количество кадров на пленке >= 24.
10. В таблице Camera выделить группы по равенству поля Producer. Найти минимальное значение среди максимумов поля Price.

**Вариант 3**

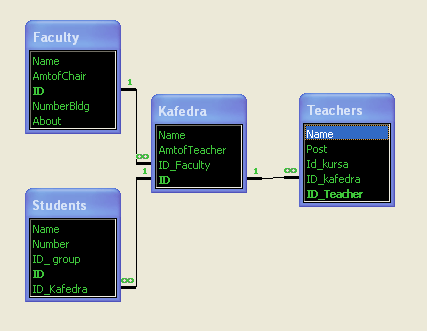
«Музыкальные диски»

****

1. Вывести средний размер дисков сделанных фирмой Звездная\_серия.
2. Вывести количество дисков с непустым полем **Prodused**.
3. Вывести количество дисков с различным размером где в описании написано «пиратская версия».
4. Вывести среднюю цену на все диски, считая пустое значение за 0.
5. Выбрать среднее значение цены для всех дисков. Сгруппировать по фирме-производителю.
6. Выбрать сумму цен всех дисков сгруппировав их по размеру и производителю.
7. Выбрать количество дисков сделанных в сентябре и ноябре.
8. Выбрать название и средние значение цены для групп дисков где средняя цена превышает 4000. Сгруппировать по фирме-производителю.
9. Выбрать название и средний размер дисков в названии которых не встречается слово Day и средний размер не превышает 560. Сгруппировать по фирме-производителю.
10. Выбрать максимальное среднее значение цены за диски сделанными разными фирмами-производителями.

**Вариант 4**

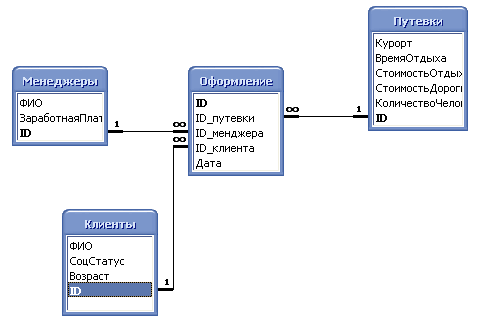
«Университет»

****

1. Вывести количество преподавателей на всех кафедрах факультета КСиС.
2. Вывести количество студентов со средним балом больше трех.
3. Вывести количество студентов у которых уже определен средний бал (значение поля не равно NULL) и стипендия больше 50000.
4. Вывести общее число среднего бала всех студентов, считая пустое значение за 0.
5. Выбрать среднее значение стипендии для всех студентов. Сгруппировать по округленному до целого среднему балу(3.23->3).
6. Выбрать сумму кафедр всех факультетов сгруппировав их по названию и адресу.
7. Вывести количество студентов поступивших летом и осенью.
8. Для каждой кафедры посчитать количество студентов с общей суммой стипендии больше 300000.
9. Выбрать группы студентов со стипендией принадлежащие к классу «Без льгот».
10. Вывести группы студентов с максимальной суммой стипендии сгруппировав их по номеру группы.

**Вариант 5**

«Турфирма»

****

1. Выведите общую сумму путевок для каждого менеджера.
2. Для определенного менеджера выведите количество клиентов, указавших свой социальный статус.
3. Выведите количество клиентов фирмы.
4. Подсчитайте среднюю стоимость дороги для всех курортов, заменяя пустые значения нолем.
5. Выведите количество клиентов каждого менеджера.
6. Для каждого менеджера выведите имя менеджера, имя и возраст самого младшего клиента.
7. Выведите градацию менеджеров в зависимости от их зарплаты: от 1000 до 800, от 750 до 500, от 450 до 250.
8. Выведите название курорта и стоимость отдыха, где средний возраст отдыхающих меньше 30 лет.
9. Для менеджера Розенкранца выведите список всех курортов, на которые он отправлял людей и самых старых клиентов, побывавших на каждом курорте.
10. Выведите минимальный средний возраст клиента для всех менеджеров.

## КОНТРОЛЬНЫЕ РАБОТЫ

### *Контрольное работа №1*

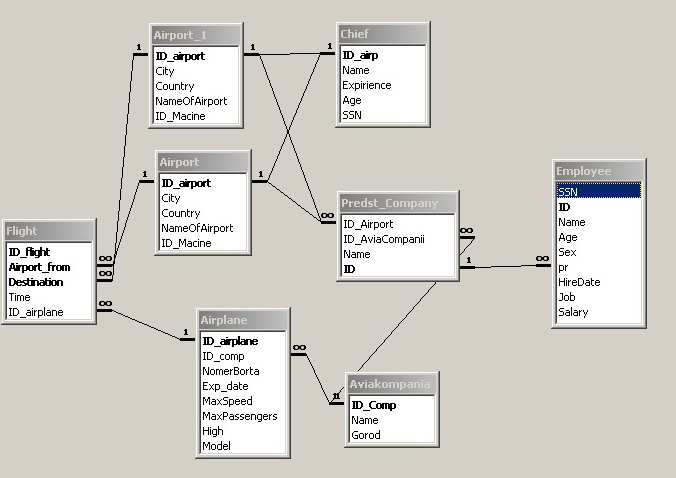
#### Часть 1.1 Основы SQL\*Plus

**Общая постановка задачи:**

Ознакомиться с возможностями среды SQL\*Plus. Ознакомиться с основными командами среды SQL\*Plus. Использовать SQL\*Plus для выполнения и редактирования запросов SQL.

**Вариант 1**

«Аэропорт»



1. Для того, чтобы загрузить SQL\*Plus, необходимо нажать кнопку «Пуск» -> «Программы»-> «Oracle – OraHome92» -> «ApplicationDevelopment» -> «SQLPlus». Выбрать соответствующий пункт меню левым щелчком мыши.

2. Появиться окно с запросом имени пользователя и пароля. Параметры входа в базу данных уточните у администратора базы данных.

3. При помощи команды, которая имеет следующий вид:

DISC[ONNECT]

Где в скобках записаны необязательные символы. Сделайте все изменения окончательными и отключитесь от системы управления базами данных Oracle, не выходя из SQL\*Plus.

4. Подключитесь к Вашей базе данных при помощи команды connect. Команда CONNECT имеет следующий вид:

CONN[ECT] username[/password][@connect\_identifier]

Вместо username, password и connect\_identifier следует использовать имя пользователя, пароль и имя базы данных.

5. Загрузите в буфер SQL\*Plus команды из файла сценария с использованием команды GET. Команда GET имеет следующий вид:

GETfile\_name[.ext]

Вместо file\_name следует подставить имя файла. Необязательным параметром является расширение файла.

6. Выведите из буфера SQL\*Plus строки SQL команды при помощи команды LIST, которая имеет следующий вид:

L[IST] [n | n m | n \* | n LAST | \* | \* n | \* LAST | LAST]

Выполните команду SQL из буфера SQL\*Plus при помощи команды RUN:

R[UN]

Очистите буфер SQL\*Plus при помощи команды, которая имеет следующий вид:

CL[EAR] BUFF[ER]

7. Запустите файл со скриптом на выполнение при помощи команды, которая имеет вид:

@ {url|file\_name[.ext]}

Где url обозначает полный путь к файлу. А file\_name соответственно обозначает название файла.

Также Вы можете использовать команду start. Она имеет следующий вид:

STA[RT] {url|file\_name[.ext]}

Команда принимает такие же аргументы, как и предыдущая.

Измените команду, которая находится в буфере SQL\*Plus при помощи команды EDIT:

ED[IT] [file\_name[.ext]]

Закройте текстовый редактор, сохранив все изменения в нем. Запустите на выполнение команду из буфера SQL\*Plus.

8. Внесите изменения в файл со скриптом при помощи команды EDIT:

ED[IT] [file\_name[.ext]]

9. Выведите описание одной из созданных Вами таблиц при помощи команды DESCRIPTION:

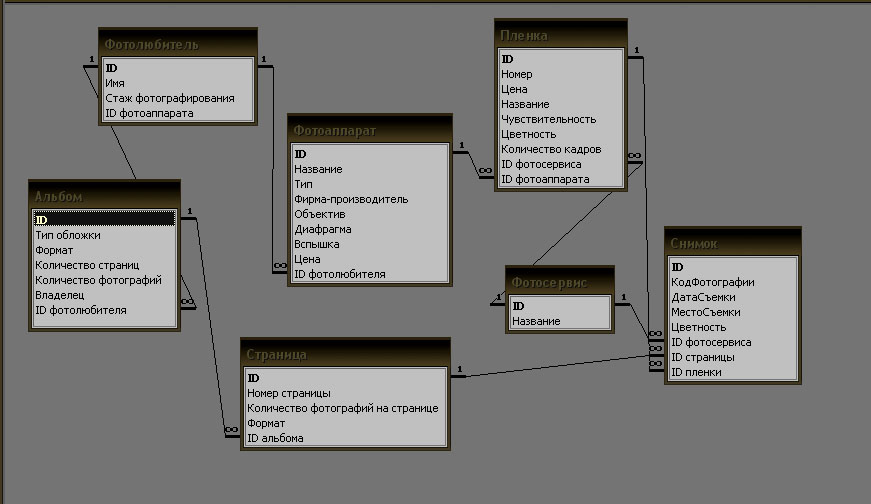
DESC[RIBE] {object[@connect\_identifier]}

Где object – это объект для описания, а connect\_identifier – имя Вашей базы данных.

10. Выйдите из SQL\*Plus, используя команду EXIT.

**Вариант 2**

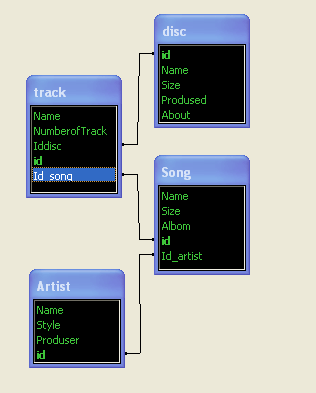
«Фотоальбом»

****

1. Запустите sqlplusw.
2. Зайдите в базу данных, используя логин, пароль и название БД, выданные преподавателем.
3. Отсоединитесь от данной базы данных.
4. Снова выполните соединение с БД.
5. Запишите содержимое файла Photoalbum.sql в буфер SQL.
6. Выведите содержимое буфера SQL. Выполните команды из буфера. Очистите буфер SQL.
7. Запустите файл Photoalbum командой @ или start. Отобразите первые 7 строк буфера SQL. Выполните команды из буфера, не отображая сами команды.
8. При помощи команды SQL\*Plus откройте Photoalbum для правки в текстовом редакторе.
9. Выведите на экран содержимое таблицы Film.
10. Завершите работу с sqlplusw.

**Вариант 3**

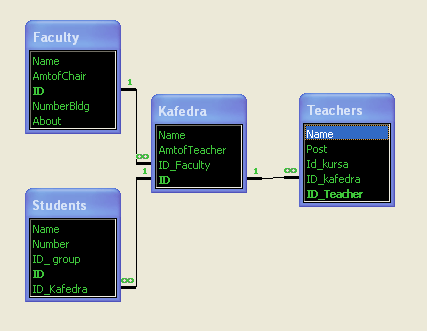
«Музыкальные диски»

****

1. Запустите sqlplusw.
2. Зайдите в базу данных, используя логин, пароль и название БД, выданные преподавателем.
3. Отсоединитесь от данной базы данных.
4. Снова выполните соединение с БД.
5. Запишите содержимое файла Photoalbum.sql в буфер SQL.
6. Выведите содержимое буфера SQL. Выполните команды из буфера. Очистите буфер SQL.
7. Запустите файл Photoalbum командой @ или start. Отобразите первые 7 строк буфера SQL. Выполните команды из буфера, не отображая сами команды.
8. При помощи команды SQL\*Plus откройте Photoalbum для правки в текстовом редакторе.
9. Выведите на экран содержимое таблицы Film.
10. Завершите работу с sqlplusw.

**Вариант 4**

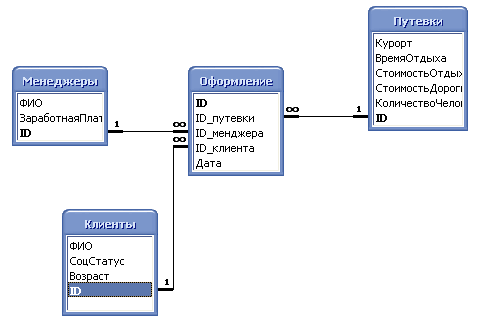
«Университет»

****

1. Запустите sqlplusw.
2. Зайдите в базу данных, используя логин, пароль и название БД, выданные преподавателем.
3. Отсоединитесь от данной базы данных.
4. Снова выполните соединение с БД.
5. Запишите содержимое файла Photoalbum.sql в буфер SQL.
6. Выведите содержимое буфера SQL. Выполните команды из буфера. Очистите буфер SQL.
7. Запустите файл Photoalbum командой @ или start. Отобразите первые 7 строк буфера SQL. Выполните команды из буфера, не отображая сами команды.
8. При помощи команды SQL\*Plus откройте Photoalbum для правки в текстовом редакторе.
9. Выведите на экран содержимое таблицы Film.
10. Завершите работу с sqlplusw.

**Вариант 5**

«Турфирма»

****

1. Запустите Oracle. Для этого выполните в командной строке (Пуск->Выполнить или Start->Run) команду sqlplusw.
2. Зайдите под своим пользователем.
3. Отсоединитесь от базы, используя команду disconnect.
4. Присоединитесь к базе, используя команду connect пользователь/пароль@база
5. Командой Get загрузите скриптовый файл «ТурФирма.sql».
6. Команда LIST служит для просмотра SQL буфера. С помощью команды RUN выполните команды, хранящиеся в буфере. CL BUF очистит буфер.
7. Вы можете выполнить скрипт хранящийся в файле командой START ТурФирма.sql либо @ ТурФирма.sql. Используйте EDIT, чтобы отредактировать и сохранить текущий буфер (в файл afiedt.buf).  / [Enter]
8. EDIT ТурФирма.sql позволяет отредактировать уже созданный скриптовый файл.
9. DESC выводит описание таблицы. Выполните DESC Клиенты.
10. Выйти из SQL\*PLUS позволяет команда EXIT.

#### Часть 1.2 SQL: простейшие запросы

**Общая постановка задачи:**

Изучить структуры запроса SELECT. Получить практические навыки в использовании следующих базовых возможностей:

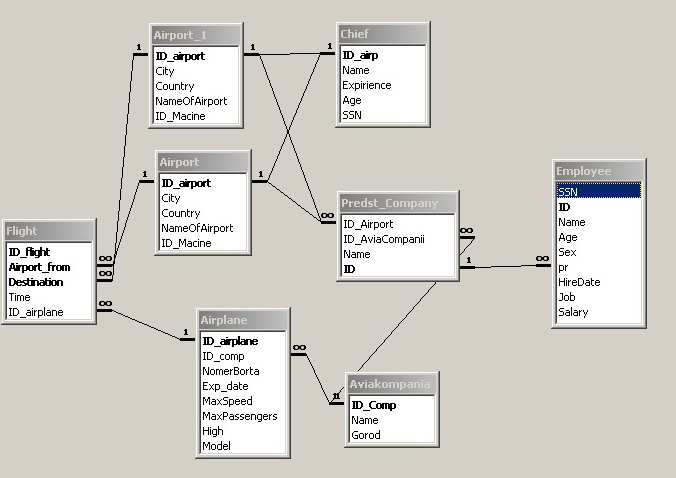
1. выборка данных из одной таблицы;

2. использование арифметических выражений и работа с символьными значениями в запросах;

3. сортировка результатов запроса.

**Вариант 1**

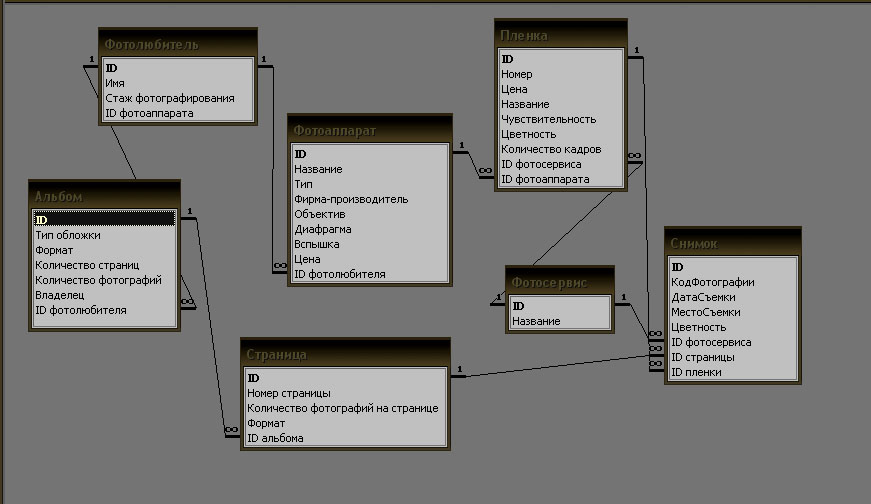
«Аэропорт»



1. Выбрать значение всех полей таблицы Employee
2. Выбрать значение полей Name, Age и Sex из таблицы Employee.
3. Из таблицы Employee вывести значение полей Name, Sex и поле, которое показывает возраст сотрудника увеличенный на 10 лет.
4. Выбрать значения полей Name, Age и Sex используя псевдонимы.
5. Выбрать значения полей Name и Age соединив их в строку типа: “[Name] is [Age] yearsold”.
6. Выбрать неповторяющиеся имена сотрудников.
7. Вывести неповторяющиеся имена и номера социального страхования сотрудников.
8. Вывести имя, возраст и пол сотрудника. Отсортировать сотрудников по имени. Сотрудников с одинаковым именем отсортировать по возрасту.

**Вариант 2**

«Фотоальбом»

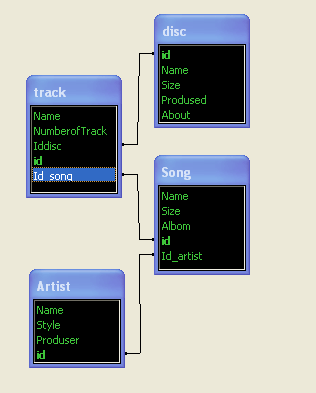
****

1. Выбрать значение всех полей таблицы Album.
2. Выбрать значение полей ID, Cover и NumberOfPages из таблицы Album.
3. Из таблицы Album вывести значение полей ID, Format и поле, которое показывает удвоенное количество фотографий + 3.
4. Выбрать значения полей Cover, NumberOfPages, используя псевдонимы Folder и Pages соответственно.
5. Выбрать значения полей ID, NumberOfPhotos таблицы Album, соединив их в строку типа: «Thealbum №[ID] contains [NumberOfPhotos] photos».
6. Выбрать неповторяющиеся значения поля Format .
7. Вывести значения полей Cover, Format. Значение поля Cover не должно повторяться.
8. Вывести строки таблицы Album, отсортировав их по выражению

NumberOfPhotos \* NumberOfPages.

**Вариант 3**

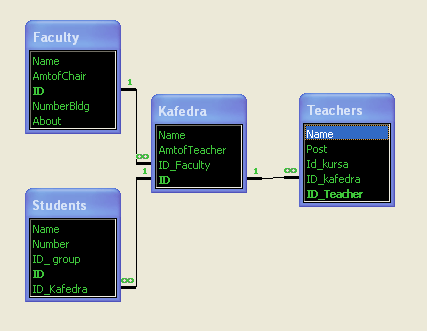
«Музыкальные диски»

****

1. Выбрать всю информацию о дисках
2. Выбрать название и размер всех дисков.
3. Выбрать название и название производителя(prodused) всех дисков, а так же размер дисков, увеличенный три раза.
4. Выбрать название и описание всех дисков, используя псевдонимы.
5. Вывести в один столбец название и размер всех дисков.
6. Выбрать неповторяющиеся размеры дисков.
7. Выбрать название и размер диска. Название и размер не должны повторятся.
8. Выбрать название, утроенный размер и производителя всех дисков отсортировав их по утроенному размеру.

**Вариант 4**

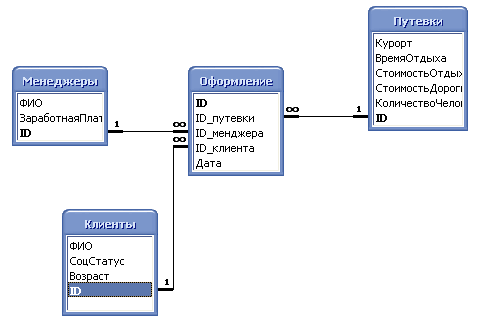
«Университет»

****

1. Выбрать всю информацию о факультетах**.**
2. Выбрать название и адрес всех факультетов**.**
3. Выбрать название и количество кафедр всех факультетов, а так же количество кафедр увеличенных в два раза.
4. Выбрать название и адрес всех факультетовиспользуя псевдонимы.
5. Выбрать название и количество кафедр всех факультетов объединив(соединив) их в одно поле.
6. Выбрать адреса факультетов находящиеся в разных корпусах**.**
7. Выбрать название и количество кафедр всех факультетов с разным количеством кафедр.
8. Выбрать название, адрес и количество кафедр всех факультетовотсортировав их по количеству кафедр и адресу.

**Вариант 5**

«Турфирма»

****

1. Выберите все столбцы из таблицы «Путевки».
2. Выберите все курорты и время отдыха на них.
3. Посчитайте стоимость одного дня отдыха на курорте. Стоимость дороги не учитывается.
4. Отобразите список клиентов в колонке «Плательщики».
5. Выведите имя клиента и его возраст в один столбец, разделив их дефисом.
6. Выберите все курорты из таблицы «Путевки», исключая повторы.
7. Выберите все курорты и время отдыха на них из таблицы «Путевки», чтобы ни одно значение в полях не повторялось.
8. Выберите все столбцы из таблицы «Клиенты», отсортировав результат по фамилии , а однофамильцев по социальному статусу.

### *Контрольное работа №2*

#### Часть 2.1 SQL: DDL

**Общая постановка задачи:**

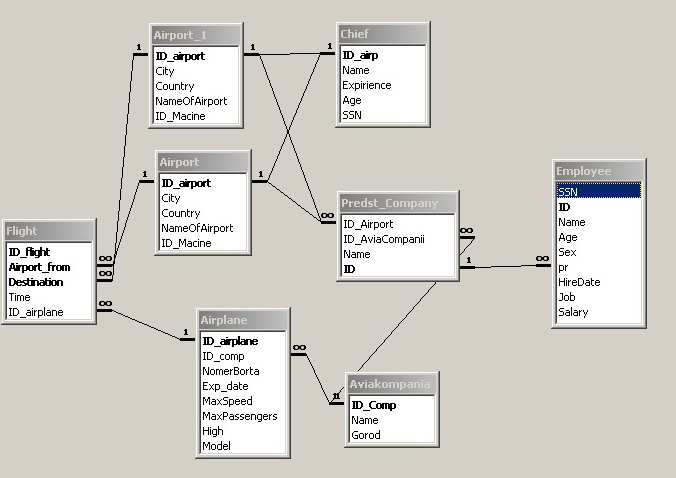
Изучить основные возможности Data Definition Language (DDL). Получить практические навыки выполнения с его помощью, следующих операций:

1. создание и удаление таблиц,

2. изменение таблиц.

**Вариант 1**

«Аэропорт»



1. Вставить в таблицу Employee три строки:

1 , NULL, Evgen, 22, Male, 1, P.Brovki,6

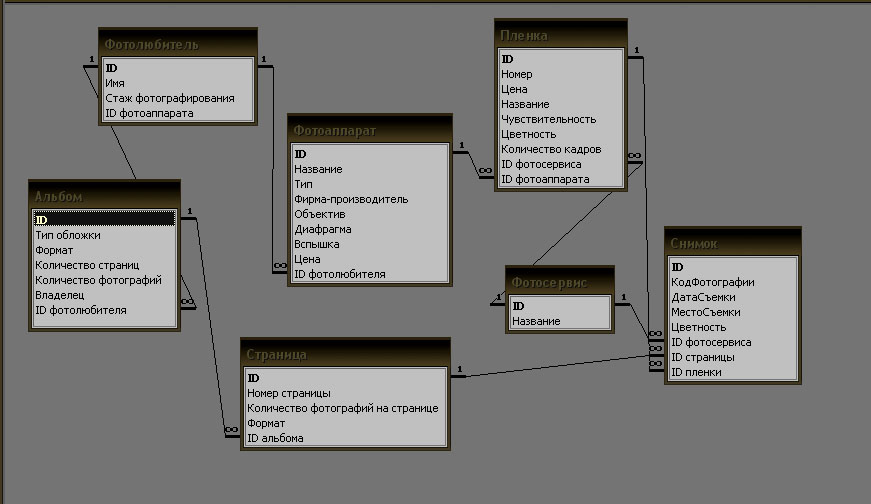
120, 2, Marina, 20, Female, 2, P.Brovki,12

13, 3, Alex, NULL, Male, 1, P.Brovki,6

1. В таблице Employee, в строках, где поле pr=2 изменить значение поле Address на «Gikalo».
2. Измените значение адреса в таблице Employee на “Gikalo,6” , где имя работника включает в себя имя Evgen.
3. Все предварительные обновления сделайте окончательными (используя команду COMMIT)
4. Удалите из таблицы Employee строку, где значение поля Name = Alex.
5. Аннулируйте предыдущее изменение таблицы, используя команду ROLLBACK.
6. Вывести все строки, где значение поля pr=1..
7. Заполните поля всех таблиц вашей БД.

**Вариант 2**

«Фотоальбом»

****

1. Создайтетаблицу Album сполями: ID, Cover, Format, NumberOfPages, NumberOfPhotos.

Для полей таблицы нужно указать, что поля ID и NumberOfPages не могут иметь неопределенное значение(NOTNULL).

2. Создайте новую таблицу Page с полями:

ID NOT NULL NUMBER(6)

Number NUMBER(3)

Format VARCHAR2(5)

Album\_ID NUMBER(2)

3. Увеличьте в таблице Album размер поля Cover на тип char с размерностью 25 и проверить, изменился ли тип поля.

4. Убедитесь, используя представление USER\_TABLES, что обе ваши таблицы Album и Page содержатся в словаре данных.

5. Создайте таблицу Album\_2, используя структуру таблицы Album.

6. Удалите таблицу Album.

7. Переименуйте таблицу Album\_2 в Album.

8. В таблице Page в качестве первичного ключа укажите поле ID,

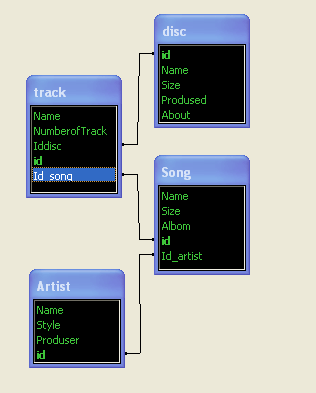
9. В таблице Page укажите внешний ключ (Album \_ID).

10. Убедитесь, что ограничения были добавлены, при помощи представления USER\_CONSTRAINTS.

11.Измените таблицу Page, добавив в нее поле Color, тип данных VARCHAR2(20).

**Вариант 3**

«Музыкальные диски»

****

1.Создать таблицу **disc** с полями **Name**, **Size**,**Crdate(**тип **DATE), Cost, Prodused**,**About**.Для полей таблицы нужно указать, что поля **Name** и **Size** не могут иметь неопределенное значение(**NOTNULL**).

2.Создайде таблицу **Track** на подобии примера показанного снизу. Вывести таблицу.

|  |  |
| --- | --- |
| Column Name | Datatype(length) |
| Name | Varchar2(5) |
| NumberofTrack | Number(3) |
| Iddisc | Number(2) |
| Id | Number(3) |

3.Изменить тип поля **Name** на тип **char** с размерностью 15 и проверить изменился ли тип поля.

4.Убедитесь что таблицы **disc** и **Track** находятся в Словаре данных(используя представление **user\_tables**).

5.Создайте таблицу **disc2** используя структуру таблицы **disc** и добавив новое поле **Id**.

6.Удалите таблицу **disc** (используя команду **Drop**).

7.Переименуйте таблицу **disc2** в **disc**.

8.В таблице **Track** укажите первичным ключом поле **id**.

9.Укажите в таблице **Track** внешний ключ (**Iddisc**).

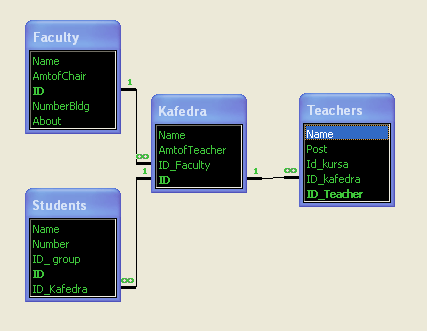
10.Проверьте установленные ограничения командой **USER\_CONSTRAINTS**.

11.Добавьте в таблицу **Track** новое поле **size**(Длина track'а) с типом **NUMBER** размерности 3.

12. Создайте оставшиеся таблицы вашей БД.

**Вариант 4**

«Университет»

****

1.Создать таблицу **Faculty** с полями **Name**, **Amtofchair**(количествоКафедр),**NumberBldg**(адрес),**About**.Для полей таблицы нужно указать, что поля Name и Amtofchair не могут иметь неопределенное значение(**NOTNULL**).

2.Создайде таблицу **Kafedra** на подобии примера показанного снизу. Вывести таблицу.

|  |  |
| --- | --- |
| Column Name | Datatype(length) |
| Name | Varchar2(10) |
| AmtofTeacher | Number(3) |
| Idfaculty | Number(2) |
| Id | Number(3) |

3.Изменить тип поля **Name** на тип **char** с размерностью 15 и проверить изменился ли тип поля.

4.Убедитесь что таблицы **Faculty** и **Kafedra** находятся в Словаре данных(используя представление **user\_tables**).

5.Создайте таблицу **Faculty2** используя структуру таблицы **Faculty** и добавив новое поле **Id**.

6.Удалите таблицу **Faculty** (используя команду **Drop**).

7.Переименуйте таблицу **Faculty2** в **Faculty**.

8.В таблице **Kafedra** укажите первичным ключом поле **id**.

9.Укажите в таблице **Kafedra** внешний ключ (**Idfaculty**).

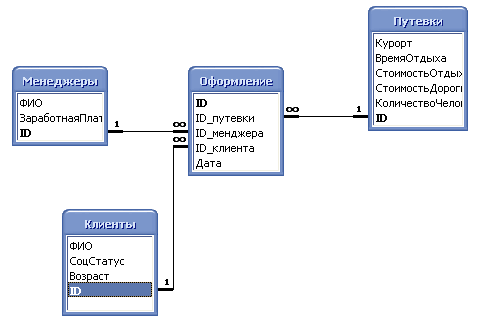
10.Проверьте установленные ограничения командой **USER\_CONSTRAINTS**.

11.Добавьте в таблицу **Kafedra** новое поле **NamberClass**(номер кабинета) с типом **NUMBER** размерности 2.

12. Создайте оставшиеся таблицы вашей БД.

**Вариант 5**

«Турфирма»

****

**«SQL: CREATE»**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Курорт | ВрмяОтдыха | СтоимостьОтдыха | СтоимостьДороги | КоличествоЧеловек | ID |
| Кипр | 10 | 600 | 200 | 2 | 1 |
| Мадагаскар | 20 | 1000 | 100 | 5 | 2 |

1. Добавьте в таблицу Путевки первую запись из таблицы используя команду INSERT и не перечисляя колонки.
2. Добавьте в таблицу Путевки вторую запись из таблицы, перечисляя колонки.
3. Подтвердите создание записей.

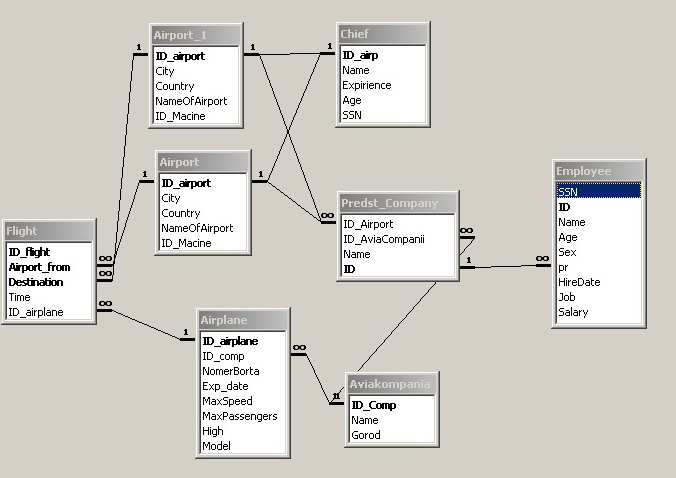
#### Часть 2.2 SQL: INSERT / UPDATE / DELETE

**Общая постановка задачи:**

Получить практические навыки вставки/удаления/модификации данных в таблицах с помощью соответствующих операторов DataManipulationLanguage.

**Вариант 1**

«Аэропорт»



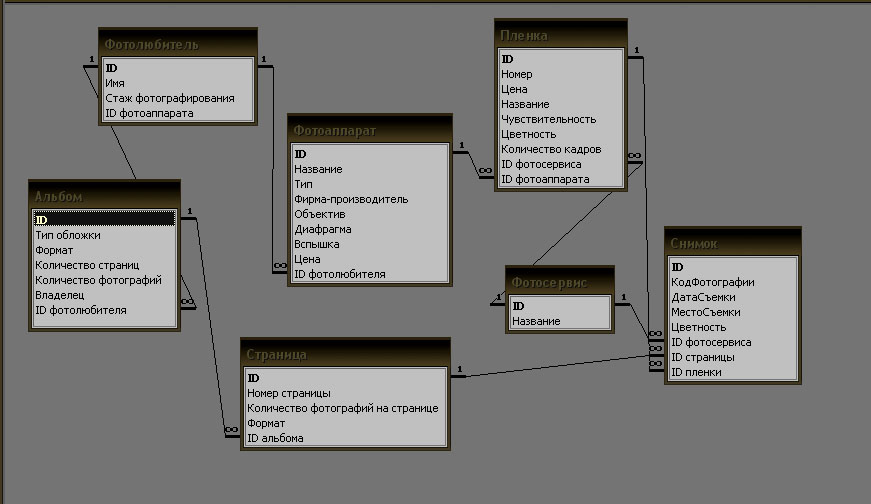
1. Создать таблицу Employee с полями SSN, ID,Name,Age,Sex,pr.Для полей таблицы нужно указать, что поля Name и Sex не могут иметь неопределенное значение (NOTNULL).
2. Создайде таблицу Chief по схеме, показанной снизу. Вывести таблицу.

|  |  |
| --- | --- |
| Column Name | Datatype(length) |
| ID\_airp | Number(5) |
| Name | Varchar2(10) |
| Experience | Number(2) |
| Age | Varchar2(6) |
| SSN | Number(10) |

1. Изменить тип поля Name на тип char с размерностью 10 и проверить изменился ли тип поля.
2. Убедитесь, что таблицы Employee и Chief находятся в DataDictionary (используя представление user\_tables).
3. Создайте таблицу Employee2, используя структуру таблицы Employee и добавив новое поле Address.
4. Удалите таблицу Employee (используя команду Drop).
5. Переименуйте таблицу Employee2 в Employee.
6. В таблице Chief укажите первичным ключом поле ID\_airp.
7. Укажите в таблице Employee внешний ключ (ID таблицы Predst\_Company).
8. Проверьте установленные ограничения, используя USER\_CONSTRAINTS.
9. Добавьте в таблицу Employee новое поле Address с типом VARCHAR2 размерности 15.

**Вариант 2**

«Фотоальбом»

****

1. Вставьте в таблицу Album строки:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Cover | Format | NumberOfPages | NumberOfPhotos |
| 1 | Leather | 10 x 15 | 60 | 3 |
| 2 | Plastic | 9 x 13 | 40 | 2 |
| 3 | Cloth | 10 x 15 | 80 | 2 |
| 4 | Leather | 13 x 18 | 120 | 1 |
| 5 | Cardboard | 10 x 15 | 80 | 1 |

Проверьте содержимое таблицы, используя команду

SELECT \* FROMAlbum.

2. В таблице Album изменить значение поля NumberOfPages на 100 в тех строках, где Format == 10 x 15.

3. Сделать количество страниц в альбомах с кожаной обложкой(Leather) таким же, как в альбомах с пластиковой (Plastic).

4. Все предварительные изменения сделайте окончательными, используя команду COMMIT.

5. Удалите из таблицы Album строку с полем Cover, равным Cardboard.

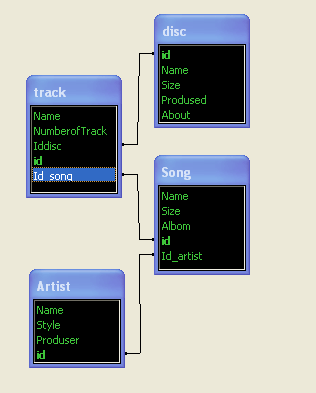
6. Отмените предыдущее действие с таблицей при помощи команды ROLLBACK.

7. Выведите все строки таблицы с полем Format, равным 10 х 15.

8. Заполните поля всех таблиц вашей БД.

**Вариант 3**

«Музыкальные диски»

****

1.Создайте таблицу **KatalogCD**c полями **NameofShop**, **NameDisc, Size, Count,Produsedby, ID**и заполните ее строками приведенными ниже:

Мистерия звука,Nightwish,598,45,newCD,1

Мистерия звука,Green\_Day,488,72,Null,2

Мистерия звука,Blink\_182,532,25,Звездная\_серия,3

Олимп,Placedo,350,12.Null,4

Цум,Guano Apes,699,30,Null,5

2.Вставьтевтаблицу**disc** строкиизтаблицы **KatalogCD**гдеполе**NameofShop** = ‘Мистериязвука’ ииспользуятолькополя**NameDisc, Size, Produsedby, ID.**

3. В таблице **disc** в строках где поле **Name =** Green\_dayизменить значение поле **About**на «Рок».

4. Вставьте в таблицу строку заполнив только поля **Name, size** и**Id.**

Korn,500,3

5. Диски Green\_Day теперь производит та же фирма что и диски Nightwish. Сделайте соответствующие изменения в таблице **disc** используя подзапрос

6. Все предварительные обновления сделайте окончательными(используя команду COMMIT)

7.Удалите из таблицы**disc**строки где значение поля **Name** совпадает со значением поля **NameDisc** таблицы **FacofUnivers**.

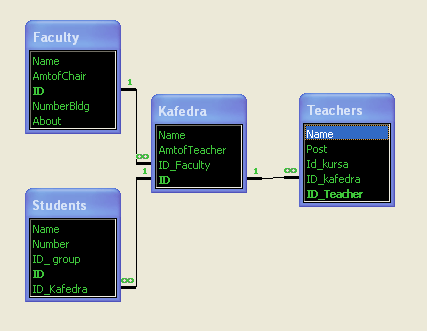
8. Аннулируйте предыдущее изменение таблицы используя команду ROLLBACK.

9.Вывести все строки где значение поля **Produsedby =** newCD**.**.

10. Заполните поля всех таблиц вашей БД(Не меньше 5)

**Вариант 4**

«Университет»

****

1.Создайте таблицу **FacofUnivers**c полями **University**, **Faculty, Amtofchair, Address, ID**и заполните ее строками приведенными ниже:

БГУиР, КСиС ,3,Бровки 4,1

БГУиР, ФиТУ,2,,Бровки 4,2

БГУиР, ФРЭ,4,Бровки 7,3

БГПА, экономика,2, Коласа 73,4

2..Вставьте в таблицу **Faculty**строки из таблицы **FacofUnivers**где поле **University**= БГУиР и используя только поля **Faculty, Amtofchair, NumberBldg, ID.**

3. В таблице **Faculty** в строках где поле **Name =**КСиСизменить значение поле **About**на «программирование».

4. Вставьте в таблицу строку заполнив только поля **Name, NumberBldg**и**Id.**

экономика, Коласа 73,4

5. Факультет ФиТУ перенесли в корпус где находиться факультет ФРЭ. Сделайте соответствующие изменения в таблице **Faculty**используя подзапрос.

6. Все предварительные обновления сделайте окончательными(используя команду COMMIT)

7.Удалите из таблицы**Faculty** строки где значение поля **NumberBldg**совпадает со значением поля **Address**таблицы **FacofUnivers**.

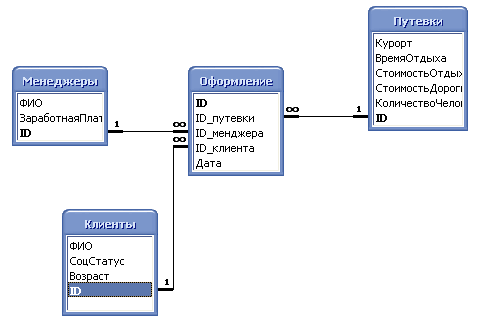
8. Аннулируйте предыдущее изменение таблицы используя команду ROLLBACK.

9.Вывести все строки где значение поля **NumberBldg =** Бровки 4**.**.

10. Заполните поля всех таблиц вашей БД(Не меньше 5)

**Вариант 5**

«Турфирма»

****

1. Создать таблицу Оформление с полями ID, ID\_путевки, ID\_менеджера, ID\_клиента, Дата.Для полей таблицы нужно указать, что поля ID, ID\_путевки, ID\_менеджера, ID\_клиента не могут иметь неопределенное значение(NOT NULL).
2. Создайде таблицу Путевки на основе шаблона приведенного ниже. Вывести таблицу.

|  |  |
| --- | --- |
| Column Name | Datatype(length) |
| Курорт | Varchar2(10) |
| ВремяОтдыха | Number(6) |
| СтоимостьОтдыха | Number(6) |
| СтоимостьДороги | Number(2) |
| КоличествоЧеловек | Number(3) |
| ID | Number(6) |

1. Изменить тип поля Курорт на тип char с размерностью 20 и проверить изменился ли тип поля.
2. Убедитесь что таблицы Оформление и Путевки находятся в Data Dictionary(используя представление user\_tables).
3. Создайте таблицу П2 по примеру таблицы Путевки.
4. Удалите таблицу П2 (используя команду Drop).
5. В таблице Путевки укажите первичным ключом поле ID.
6. Укажите в таблице Оформление внешний ключ (ID\_путевки).
7. Проверьте установленные ограничения командой USER\_CONSTRAINTS.
8. (SELECT constraint\_name, constraint\_type,
9. search\_condition
10. FROM user\_constraints
11. WHERE table\_name = <имятаблицы>;)
12. Добавьте в таблицу Путевки новое поле Средняя\_t с типом NUMBER размерности 3.
13. Создайте все оставшиеся таблицы и установите связи между ними

## Задание на курсовую работу, ее характеристика.

Курсовая работа учебным планом не предусмотрена.