

Документ подписан простой электронной подписью

Информация о владельце:

ФИО: Таныгин Максим Олегович

Должность: и.о. декана факультета фундаментальной и прикладной информатики

Дата подписания: 21.09.2023 12:44:06

Уникальный программный ключ:

65ab2aa0d384efe8480e6a4c688eddbc475e411a

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования «Юго-Западный государственный университет» (ЮЗГУ)

Кафедра программной инженерии



УТВЕРЖДАЮ:
проректор по учебной работе
Локтионова О.Г.
2023 г.

Реализация запросов к базе данных средствами языка SQL

Методические указания к лабораторной работе
по дисциплине «Базы данных»
для студентов направления подготовки
09.03.04 «Программная инженерия»

Курск 2023

УДК 004.65

Составитель Е.И.Аникина

Рецензент

Кандидат технических наук, доцент кафедры программной инженерии *Е.А.Петрик*

Реализация запросов к базе данных средствами языка SQL: методические указания к лабораторным работам по дисциплине «Базы данных» для студентов направления подготовки 09.03.04 «Программная инженерия»/Юго-Зап. гос. ун-т; сост. Е.И. Аникина. Курск, 2023. 13 с.

Содержит задания к лабораторной работе, теоретические сведения и примеры решения задач по теме курса, связанной с изучением основ языка SQL, проектированием и реализацией запросов к базе данных средствами языка SQL.

Предназначено для студентов направления подготовки бакалавров 09.03.04 «Программная инженерия».

Текст печатается в авторской редакции.

Подписано в печать . Формат 60x84 1/16.
Усл. печ. л. . Уч.-изд. л. . Тираж 100 экз. Заказ .
Бесплатно.

Юго-Западный государственный университет
305040, Курск, ул.50 лет Октября, 94.

РЕАЛИЗАЦИЯ ЗАПРОСОВ К БАЗЕ ДАННЫХ СРЕДСТВАМИ ЯЗЫКА SQL

ЗАДАНИЕ

1. Заполнить таблицы данными (не менее пяти строк в каждой таблице). В таблицах с внешними ключами должно быть по несколько записей с одинаковыми значениями внешних ключей - показать, что реализуется отношение 1:N.

2. ИЗУЧИТЬ материал темы 6 курса лекций по *ЯЗЫКУ SQL*.

3. Спроектировать и записать на **SQL** 9 запросов следующего характера:

1. Поисковый запрос на полное совпадение по заданному критерию отбора с выводом значений из определенных полей таблицы

2. Поиск по неполному совпадению (с использованием функции Like)

3. Поиск данных, находящихся в заданном диапазоне дат.

4. Сложное условие: одновременное выполнение нескольких условий (использовать функции AND или OR)

5. Поиск по двум связанным таблицам

6. Запрос с Вычисляемым полем

7. Групповой запрос (использование обобщающих функций Sum, Average)

8. Запрос с сортировкой результатов выборки

9. Запрос с исключением из результата повторяющихся строк (distinct)

Во всех командах SQL использовать псевдонимы полей на РУССКОМ ЯЗЫКЕ.

4. Изучить технологию работы с запросами средствами ORACLE APPLICATION EXPRESS SQL WORKSHOP (документ ORACLE-APPLICATION-EXPRESS-SQL-WORKSHOP-GUIDE)

5. Реализовать **команды SQL** средствами ORACLE APPLICATION EXPRESS SQL WORKSHOP

Содержание отчета

Для каждого запроса в отчете надо привести :

1) формулировку запроса на русском языке

2) скриншот: SQL-запрос в Oracle APEX SQL Workshop

3) скриншот: результата выполнения запроса

ТЕОРИЯ

Создание запросов на выборку данных на языке SQL

Синтаксис оператора SELECT для выборки всех записей таблицы

SELECT [DISTINCT] * столбец [псевдоним] , ... FROM таблица ;	
SELECT	Список из одного или нескольких столбцов
DISTINCT	Устраняет отображения повторяющихся строк
*	Выбор всех столбцов
столбец	Имя выбираемого столбца
псевдоним	Дает возможность настроить заголовок выбранного столбца
таблица	Определяет таблицу или таблицы, содержащие столбцы

Вы можете указать псевдоним после имени столбца в списке выбора, используя пробел в качестве разделителя. Если псевдоним содержит пробелы или специальные символы, такие как знак номера # или знак доллара \$, или, если он чувствителен к регистру, заключите псевдоним в кавычки ". Пример:

```
SELECT employee_id "Табельный номер сотрудника",  
       last_name "Фамилия сотрудника",  
       first_name "Имя сотрудника"  
FROM employees
```

Ограничение выборки строк

Вы можете ограничить количество строк, которые извлекаются из базы данных, с помощью инструкции WHERE в операторе SQL. Добавляя инструкцию WHERE, вы можете задать условие, которое должно быть выполнено, и только те строки, которые соответствуют условию, будут возвращены.

При использовании инструкции WHERE:

инструкция WHERE непосредственно следует за инструкцией FROM в синтаксисе оператора SQL; инструкция WHERE состоит из ключевого слова WHERE и условия (или нескольких условий); условие инструкции WHERE указывает сравнение значений, которые ограничивают количество строк, возвращаемых запросом.

Рассмотрим базу данных, которая моделирует сдачу сессии в некотором учебном заведении. Пусть она состоит из трех отношений **R1**, **R2**, **R3**. Будем считать, что они представлены таблицами **R1**, **R2** и **R3** соответственно.

R1 = (ФИО, Дисциплина, Оценка) ; R2 = (ФИО, Группа) ; R3 = (Группы, Дисциплина)

R1		
ФИО	Дисциплина	Оценка
Петров Ф. И.	Базы данных	5
Сидоров К. А.	Базы данных	4
Миронов А. В.	Базы данных	2
Степанова К. Е.	Базы данных	2
Крылова Т. С.	Базы данных	5
Сидоров К. А.	Теория информации	4
Степанова К. Е.	Теория информации	2
Крылова Т. С.	Теория информации	5
Миронов А. В.	Теория информации	Null
Владимиров В. А.	Базы данных	5
Трофимов П. А.	Сети и телекоммуникации	4
Иванова Е. А.	Сети и телекоммуникации	5

Уткина Н. В.	Сети и телекоммуникации	5
Владимиров В. А.	Английский язык	4
Трофимов П. А.	Английский язык	5
Иванова Е. А.	Английский язык	3
Петров Ф. И.	Английский язык	5

R2	
ФИО	Группа
Петров Ф. И.	4906
Сидоров К. А.	4906
Миронов А. В.	4906
Крылова Т. С.	4906
Владимиров В. А.	4906
Степанова К. Е.	4906
Трофимов П. А.	4807
Иванова Е. А.	4807
Уткина Н. В.	4807

R3	
Группа	Дисциплина
4906	Базы данных
4906	Теория информации
4906	Английский язык
4807	Английский язык
4807	Сети и телекоммуникации

Приведем несколько примеров использования оператора **SELECT**.

- Вывести список всех групп (без повторений), где должны пройти экзамены.

SELECT DISTINCT Группы FROM R3

Результат:

Группа
4906
4807

- Вывести список студентов, которые сдали экзамен по дисциплине "Базы данных" на "отлично".

```
SELECT ФИО
```

```
FROM R1
```

```
WHERE Дисциплина = "Базы данных" AND Оценка = 5
```

Результат:

ФИО
Петров Ф. И.
Крылова Т. С.
Владимирова В. А.

- Вывести список всех студентов, которым надо сдавать экзамены с указанием названий дисциплин, по которым должны проводиться эти экзамены.

```
SELECT ФИО,Дисциплина
```

```
FROM R2,R3
```

```
WHERE R2.Группа = R3.Группа;
```

Здесь часть **WHERE** задает условия *соединения отношений R2 и R3*, при отсутствии условий соединения в части **WHERE** результат будет эквивалентен расширенному декартову произведению, и в этом случае каждому студенту были бы приписаны все дисциплины из отношения **R3**, а не те, которые должна сдавать его группа.

Результат:

ФИО	Дисциплина
Петров Ф. И.	Базы данных

Сидоров К. А.	Базы данных
Миронов А. В.	Базы данных
Степанова К. Е.	Базы данных
Крылова Т. С.	Базы данных
Владимиров В. А.	Базы данных
Петров Ф. И.	Теория информации
Сидоров К. А.	Теория информации
Миронов А. В.	Теория информации
Степанова К. Е.	Теория информации
Крылова Т. С.	Теория информации
Владимиров В. А.	Теория информации
Петров Ф. И.	Английский язык
Сидоров К. А.	Английский язык
Миронов А. В.	Английский язык
Степанова К. Е.	Английский язык
Крылова Т. С.	Английский язык
Владимиров В. А.	Английский язык
Трофимов П. А.	Сети и телекоммуникации
Иванова Е. А.	Сети и телекоммуникации
Уткина Н. В.	Сети и телекоммуникации
Трофимов П. А.	Английский язык
Иванова Е. А.	Английский язык
Уткина Н. В.	Английский язык

- Вывести список лентяев, имеющих несколько двоек.

```

SELECT DISTINCT R1.ФИО
FROM R1 a, R1 b
WHERE a.ФИО = b.ФИО AND
a.Дисциплина <> b.Дисциплина AND
a.Оценка <= 2 AND b.Оценка <= 2;

```

Здесь мы использовали псевдонимы для именования отношения **R1 a** и **b**, так как для записи условий поиска нам необходимо работать сразу с двумя экземплярами данного отношения.

Результат:

ФИО
Степанова К. Е.

Из этих примеров хорошо видно, что логика работы оператора выбора (*декартово произведение—селекция—проекция*) не совпадает с порядком описания в нем данных (сначала *список* полей для проекции, потом *список* таблиц для декартова произведения, потом условие соединения). Дело в том, что *SQL* изначально разрабатывался для применения конечными пользователями, и его стремились сделать возможно ближе к языку естественному, а не к языку алгоритмическому. По этой причине *SQL* на первых порах вызывает путаницу и раздражение у начинающих его изучать профессиональных программистов, которые привыкли разговаривать с машиной именно на алгоритмических языках.

Наличие неопределенных (**Null**) значений повышает гибкость обработки информации, хранящейся в *БД*. В наших примерах мы можем предположить ситуацию, когда студент пришел на экзамен, но не сдавал его *по* некоторой причине, в этом случае оценка *по* некоторой дисциплине для данного студента имеет неопределенное значение. В данной ситуации можно поставить вопрос: "Найти студентов, пришедших на экзамен, но не сдававших его с указанием названия дисциплины". Оператор **SELECT** будет выглядеть следующим образом:

```
SELECT ФИО, Дисциплина
FROM R1
WHERE Оценка IS NULL
```

Результат:

ФИО	Дисциплина
Миронов А. В.	Теория информации

Групповые операции

Групповые операции обрабатывают по нескольку строк и возвращают один обобщенный результат. Фактически они позволяют объединять в той или иной форме сходные элементы информации, возвращаемые из базы данных. Для выполнения групповых операций Oracle предоставляет обобщающие функции и инструкции группировки (GROUP BY, HAVING и другие).

Синтаксис большинства обобщающих функций:

```
обобщающая_функция ([DISTINCT | ALL] выражение)
```

Обобщающая_функция – указывает имя обобщающей функции: COUNT (количество), AVG (среднее значение), MAX (максимальное значение), MIN (минимальное значение), STDDEV (стандартное среднеквадратичное отклонение), SUM (сумма), VARIANCE (статистическая **дисперсия**).

DISTINCT – указывает, что обобщающая функция должна учитывать только неповторяющиеся значения выражения.

ALL – указывает, что обобщающая функция должна учитывать все значения выражения, в том числе и дублирующиеся. По умолчанию считается, что использовано ALL.

Выражение – указывает столбец или любое другое выражение, по которому необходимо выполнить обобщение.

Значения NULL для выражения обобщающими функциями игнорируются, в результат не входят.

Примеры

Select MAX(salary) from Employees – найти максимальное значение по столбцу salary во всей таблице EMPLOYEES.

Select count(*) from Regions – вычисляет количество записей в таблице REGIONS.

Инструкция GROUP BY

Используется совместно с обобщающими функциями, разбивает результирующее множество на несколько групп, а затем для каждой группы выдается одна строка сводной информации. Если в списке SELECT присутствует смесь обобщенных и необобщенных значений, SQL считает, что нужно выполнить операцию Group by, поэтому все необобщенные выражения должны быть указаны и в инструкции Group by. Если этого не сделать, Oracle выдаст сообщение об ошибке.

Не разрешено использование групповой (обобщающей) функции в инструкции GROUP BY.

При группировке по столбцу, содержащему в некоторых строках NULL-значения, все строки с NULL-значениями помещаются в одну группу и представляются в выводе одной сводной строкой.

Для получения итоговых результатов с использованием инструкции GROUP BY существует возможность фильтрации записей таблицы при помощи инструкции WHERE. При выполнении оператора SQL, содержащего инструкции WHERE

и GROUP BY, Oracle сначала применяет инструкцию WHERE и отсеивает строки, не удовлетворяющие условию WHERE. Затем строки, удовлетворяющие условию WHERE, группируются в соответствии с инструкцией GROUP BY. Синтаксис SQL требует, чтобы инструкция WHERE предшествовала инструкции GROUP BY.

Объединение таблиц

Иногда возникает необходимость отображать данные из нескольких таблиц. Чтобы сделать это, в инструкции FROM оператора SELECT указывается список имен таблиц, из которых извлекаются данные. Если информация поступает из более чем одной таблицы, происходит объединение таблиц.

Например, в таблице EMPLOYEES столбец DEPARTMENT_ID представляет номер отдела сотрудника. В таблице DEPARTMENTS есть столбец DEPARTMENT_ID, а также столбец DEPARTMENT_NAME. Можно объединить данные из таблиц EMPLOYEES и DEPARTMENTS с использованием столбца DEPARTMENT_ID и подготовить отчет, который будет показывать имена работников и названия отделов.