ИСПОЛЬЗОВАНИЕ СТРУКУТРИРОВАННОГО ЯЗЫКА ЗАПРОСОВ

Лабораторный практикум по Базам данных, занятие 3 «Выбор данных из нескольких источников»

МГУ им. А.А. Кулешова, ФМЕ, 1 курс 2 семестр
заочная форма получения образования
специальность ИПОВС, 2020-2021 учебный год
доцент Сидоренко И.Н.

1 Цель работы

Создать базу данных и несколько таблиц. Подготовить и реализовать серию запросов, связанных с выборкой информации из созданных таблиц. Изучить функции для работы с датой и временем. Подготовить и реализовать серию запросов, связанных с выборкой информации из таблиц.

Содержание работы

1. *Создать базу данных «Книги» (**B_XX_ГГ_YY)** , (где XX-номер группы, ГГ –год группы, YY-номер в журнале)*
2. *Создание таблиц в базе данных и заполнение их записями.*
3. *Создать в БД **B_XX_ГГ_YY** таблицу авторов **Avt_XX_ГГ_YY**:*

Avt_id	*A_firstname (varchar50)*	*A_lastname (varchar40)*	*BirthD (date)*	*Sex (enum)*	*Town (int)*
1	*Иванов*	*Иван*	*15.09.1989*	*м*	*2*

4. *Проверить, что таблица была создана (SHOW TABLES);*
5. *Заполнить ее записями - придумать штук 10 авторов разного пола, возраста от 30 до 80 лет и с разным количеством детей.*
6. *Вывести содержимое таблицы.*
7. *Создать таблицу книги **Book_XX_ГГ_YY**,*
8. *Задать **значение по умолчанию** для поля **Student - XXГГYY** (XX-номер группы, ГГ-год группы, YY-номер в журнале)*
9. *Добавить для столбца Avt_id ограничение **Внешний ключ**:*

Book_id	*Book_name*	*Avt_id*	*Student*
Int unsigned not null primary key auto_increment	*varchar(10)*	*Int, unsigned*	*int*

10. *Int, unsigned, FOREIGN KEY (Avt_id) REFERENCES Avt (Avt_id)*
11. *Заполнить ее любым способом. {Авторы берутся из предыдущей таблицы). Могут быть авторы, у которых нет книг.*
12. *Посмотреть структуру таблиц (DESCRIBE). Вывести содержимое таблицы.*

Выполнение запросов.

13. *Вывести авторов, родившихся в конкретном месяце (например, в феврале);*
14. *Вывести фамилии авторов, у которых есть книги;*

F	***m***
5	*6*

15. *Вывести количество авторов каждого пола (2 варианта вывода)*

F	*5*
m	*6*

16. *Вывести авторов, фамилия которых содержит букву «В».*
17. *Вывести авторов, которым меньше 50 лет.*
18. *Вывести 5 самых старых авторов.*
19. *Вывести общее количество книг у авторов каждого пола.*
20. *Вычислить, сколько лет автору на сегодняшний день (использовать текущую дату), вывести имя автора, дату рождения и возраст.*
21. *Вывести все названия книг одного автора.*
22. *Вывести список авторов и количество книг, которых они написали.*
23. *Вывести автора, у которого наибольшее количество книг.*
24. *Используя объединение таблиц, вывести названия книги, имя автора, дату рождения автора сначала для авторов, родившихся в январе, затем в феврале.*
25. *Вывести название книг, фамилию автора и название города, для авторов из одного города (на ваш выбор), используя вложенный запрос.*

Методические указания к выполнению лабораторной работы №3.

Выбор данных из значений типа Date.

DAY(date) и DAYOFMONTH(date) функции-синонимы, возвращают из даты порядковый номер дня месяца.

SELECT DAY('2011-04-17'), DAYOFMONTH('2011-04-17');

MONTH(date) и MONTHNAME(date) обе функции возвращают значения месяца. Первая - его числовое значение (от 1 до 12), вторая - название месяца. Названия месяцев различают регистр символов. Поэтому January будет работать, а JANUARY не будет

YEAR(date) функция возвращает значение года (от 1000 до 9999).

SELECT YEAR('2011-04-17');

DAYOFYEAR(date) возвращает порядковый номер дня в году (от 1 до 366).

ADDDATE(date, INTERVAL value) Функция возвращает дату date, к которой прибавлено значение value. Значение value может быть отрицательным, тогда итоговая дата уменьшится. В качестве значения value могут выступать не только дни, но и недели (WEEK), месяцы (MONTH), кварталы (QUARTER) и годы (YEAR).

PERIOD_ADD(period, n) функция добавляет n месяцев к значению даты period. Нюанс: значение даты должно быть представлено в формате YYYYMM. Давайте к февралю 2011 (201102) прибавим 2 месяца: SELECT PERIOD_ADD(201102, 2);.

TIMEDIFF(date1, date2) вычисляет разницу в часах, минутах и секундах между двумя датами.

PERIOD_DIFF(period1, period2) функция вычисляет разницу в месяцах между двумя датами, представленными в формате YYYYMM.

SUBTIME(date, time) функция вычитает из времени date время time:

DATE(datetime) возвращает дату, отсекая время.

TIME(datetime) возвращает время, отсекая дату.

TO_DAYS(date) и FROM_DAYS(n) взаимообратные функции. Первая преобразует дату в количество дней, прошедших с нулевого года. Вторая, наоборот, принимает число дней, прошедших с нулевого года и преобразует их в дату:

Текущую дату, месяц и год можно вывести с помощью аргумента **CURRENT_DATE** предложений **DAYOFMONTH(), MONTH() и YEAR(),**

CURDATE(), CURTIME() и NOW() Первая функция возвращает текущую дату, вторая - текущее время, а третья - текущую дату и время. Функции CURDATE() и NOW() удобно использовать для добавления в базу данных записей, использующих текущее время

Строковые функции

CHAR_LENGTH(строка); CHARACTER_LENGTH(строка)

Возвращает длину строки строка, измеренную в символах. Многобайтные символы считаются как один

CONCAT(строка1, строка2, ...)

Возвращает строку, которая состоит из сцепленных аргументов. Возвращает NULL, если любой из аргументов равен NULL. Принимает один или более аргументов. Числовой аргумент преобразуется в эквивалентную строковую форму.

FIELD(строка, строка1, строка2, строка3, ...)

Возвращает позицию вхождения аргумента строка в список строка1, строка2, строка3, ... Возвращает 0, если вхождение не найдено.

FIND_IN_SET(строка, список_строк)

Возвращает значение от 1 до N, если строка находится в списке строк список_строк, состоящего из N подстрок. Список строк - это строка, состоящая из подстрок, разделенных символом ','. Возвращает 0, если строка не входит в список строк, или если список_строк — пустая строка.

INSERT(строка, позиция, длина, новая_строка)

Возвращает строку строка, в которой подстрока длиной длина, начинающаяся с позиции позиция, заменяется строкой «новая_строка».

LEFT(строка, длина)

Возвращает первые длина символов строки строка.

LENGTH(строка)

Возвращает длину строки строка в байтах.

LOWER(строка); (UPPER(строка);)

Возвращает строку строка, в которой все символы приведены к нижнему (верхнему)регистру в соответствии с текущим набором символов.

LTRIM(строка)

Возвращает строку строка с удаленными ведущими пробелами.

REPLACE(строка, строка_2, строка_3)

Возвращает строку строка, в которой все вхождения строка_2 заменены на строка_3.

Функции сравнения строк.

Обычно, если любое выражение в сравнении строк чувствительно к регистру, то сравнение также чувствительно к регистру.

выражение LIKE шаблон [ESCAPE 'символ-отмены']

Проверка на соответствие шаблону, заданному простыми регулярными выражениями SQL. Возвращает 1 (TRUE) или 0 (FALSE). Если выражение или шаблон равны NULL, возвращает NULL.

В шаблонах LIKE можно использовать следующие два символа:

"%" – соответствие любому числу символов, включая нуль символов, "_" – соответствие любому одному символу.

-> 1

Соединения.

Соединения предназначены для обеспечения выборки данных из нескольких таблиц и включения этих данных в один результирующий набор. **Существует четыре вида соединений**: внутреннее, внешнее, полное, перекрестное.

Для объединения трех и более таблиц можно применять последовательность соединений.

Для соединения таблиц необходимо раздел FROM дополнить ключевыми словами JOIN, которое определяет соединяемые таблицы и метод соединения, и ON, указывающее общие для таблиц поля.

SELECT <select list> FROM <first table> [<alias>]
<JOIN TYPE> <second table> [<alias>] [ON <join condition>][;]

При внутреннем соединении сравниваются значения общих полей двух таблиц. Конструкция **INNER JOIN** возвращает только строки, согласованные по всем полям, которые обозначены как используемые для соединения. **Записи, для которых не имеется пары в связанной таблице, в результат не включаются.**

По умолчанию применяются внутренние соединения, ключевое слово **INNER** является необязательным

Внешние соединения LEFT OUTER JOIN, RIGHT OUTER JOIN

При использовании конструкции **OUTER** существует возможность включить в результирующий набор строки, которые соответствуют **хотя бы одному из заданных критериев.** Такие соединения применяются для получения **полного набора записей** одной из таблиц.

В соединениях различаются стороны – **левая и правая**. **Левой** считается таблица, указанная **в первую очередь**, а **правой** – таблица, **указанная после нее. Ключевое слово OUTER необязательно.**

Полные соединения FULL JOIN

Если применяется соединение с ключевым словом **FULL,** то в результаты должны быть **включены все строки из таблиц**, находящихся по обе стороны от ключевого слова **JOIN**, без каких либо исключений

Перекрестные соединения

При таком соединении выводятся **все комбинации записей таблиц**, при этом не требуется указание совпадающих значений полей, поэтому **условие ON опускается**.

Происходит соединение каждой строки таблиц, находящихся с одной стороны от ключевого слова JOIN, с каждой строкой таблиц, находящихся с другой стороны от ключевого слова JOIN. В итоге формируется **декартово произведение всех строк**, заданных по обе стороны ключевого слова JOIN.

Использование UNION

Иногда бывает нужно получить два списка записей из таблиц в виде одного. Для этой цели может быть использовано ключевое слово **UNION, которое позволяет объединить результирующие наборы данных двух запросов в один набор данных.**

Технических ограничений на количество запросов в операторе UNION не существует. Общий синтаксис следующий.

< инструкция **SELECT 1**> **UNION** [ALL | DISTINCT]
< инструкция **SELECT 2**> **UNION** [ALL | DISTINCT]

UNION -показывает, что результирующие наборы будут объединены в один результирующий набор. Дубликаты строк по умолчанию удаляются.

ALL -объединяются и дубликаты строк из всех результирующих наборов.

DISTINCT -Из результирующего набора удаляются дубликаты строк. Столбцы, содержащие значения NULL, считаются дубликатами. По умолчанию принимается DISTINCT.

Существует лишь одно важное правило, о котором следует помнить при использовании оператора UNION: **порядок, количество и тип данных столбцов должны быть идентичны во всех запросах**.

Типы данных не обязательно должны быть идентичны, но они должны быть совместимы. Например, типы CHAR и VARCHAR являются совместимыми. По умолчанию результирующий набор использует размер наибольшего из совместимых типов.

Имена полей итогового набора берутся только из первого запроса, поэтому создание псевдонимов полей выполняется в нем

Для получения отсортированного набора данных раздел **ORDER BY** указывается после последнего оператора **SELECT. В отличие от соединения, записи в итоговый набор добавляются друг за другом.**

Вложенные запросы

В SQL предусмотрена возможность объединять запросы в один путем превращения одного из них в подзапрос (вложенный запрос).

В одном запросе может быть несколько подзапросов, синтаксис у такого запроса следующий:

SELECT имя_столбца FROM Табл. WHERE часть условия IN
(SELECT имя_столбца FROM Табл WHERE часть условия IN
(SELECT имя_столбца FROM имя_таблицы WHERE условие))

Подзапросы могут выбирать только один столбец, значения которого они будут возвращать внешнему запросу. Попытка выбрать несколько столбцов приведет к ошибке. Стоит отметить, что не рекомендуется создавать запросы со степенью вложения больше трех. Это приводит к увеличению времени выполнения и к сложности восприятия кода.

Приведенный синтаксис вложенных запросов, скорее наиболее употребительный, но вовсе не единственный. Мы можем использовать любые операторы, используемые с ключевым словом WHERE.

Контрольные вопросы

1. Перечислите четыре вида соединений.
2. При каком соединении записи, для которых не имеется пары в связанной таблице, в результат не включаются?
3. При каком соединении условие ON опускается?
4. Какие соединения применяются для получения **полного набора записей** одной из таблиц?
5. С помощью какого соединения можно получить декартово произведение таблиц?
6. Чем отличается порядок записей в итоговом наборе, полученном с помощью соединения Join и объединения Union?
7. Из какого запроса берутся имена полей итогового набора при объединении (Union)?
8. Какие функции выделяют из даты значение года или месяца?
9. Какие операторы используются для определения наличия значение NULL?
10. Результат SELECT CONCAT('My', NULL, 'QL');?
11. Какое значение вернет CHAR_LENGTH('') для строки, состоящей из пяти двухбайтных символов?
12. Выведите первые пять символов строки «abcdefg»?
13. Какая функция приводит символы к нижнему регистру?
14. Выведите фамилию Иванов из строки ‘ФИО_ИВАНОВ’
15. Чем отличаются функции CHAR_LENGTH(строка) и LENGTH(строка)?
16. С помощью какой функции проверить идентичность строк?
17. Результат выполнения функции Select Month(‘2015-07-08’);
18. Как вывести сегодняшнюю дату?

Содержание отчета (в электронном виде загрузить в Moodle)

1. Тема работы
2. Цель работы
3. Выполненное задание с кодом и скриншотами.
4. Ответы на контрольные вопросы

Оформление отчета должно удовлетворять требованиям предъявляемым к курсовым работам (правила оформления курсовых смотрите на сайте факультета)