Указания к лабораторным работам курса «СУБД Oracle» (Oracle Academy)

v.1.3 09.09.2015

Составитель:

ст. преподаватель каф. ТИ

ЯрГУ им. П.Г. Демидова

Горбунов О.Е.

Введение

Лабораторные работы основываются на курсах, предоставленных в рамках программы Oracle Academy Advanced Computer Science, курсы «Oracle Database: Fundamentals I» и «Oracle Database: Fundamentals II». В квадратных скобках приводится номер раздела или задания из оригинального курса.

Все лабораторные работы проводятся в СУБД Oracle Database 11g XE тестовой базе данных HR (Human Resources), которая поставляется компанией Oracle и может быть установлена вместе с указанной СУБД.

Все манипуляции с базой данных рекомендуется производить с помощью Oracle SQL Developer, доступного на сайте компании Oracle.

База данных HR

База данных HR (Human Resources) разработана для хранения информации о сотрудниках некоторой организации. Схема базы данных приведена на рис. 1.

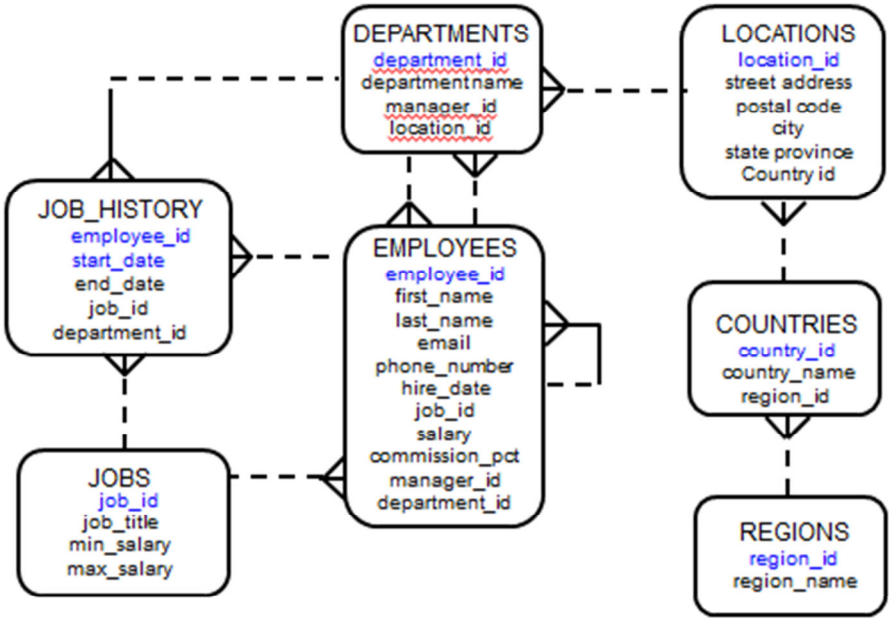

Рис. 1. Схема базы данных HR

Перед началом работы рекомендуется ознакомиться с таблицами, созданными для данной схемы. В данном курсе будут использоваться следующие талицы:

REGIONS. Информация о частях света (Азия, Африка и т.п.);

COUNTRIES. Страны. Каждая страна расположена в некоторой части света;

LOCATIONS. Адреса подразделений компании (офис, склад, производственная площадка и т.п.);

DEPARTMENTS. Отделы. В отделе могут работать сотрудники компании. У отдела может быть руководитель из числа сотрудников компании;

EMPLOYEES. Сотрудники;

JOBS. Работы. Каждый сотрудник может быть назначен на некоторую работу (должность);

JOB_HISTORY. История работ сотрудников (занимаемых должностей). Каждая работа привязана к некоторому отделу и некоторому временному интервалу.

Использование SELECT. Основы. [II]

Задание 1. [5]

Написать запрос, выводящий для каждого сотрудника следующие поля: Фамилия, JOB_ID, дата приема на работу (с псевдонимом STARTDATE), код сотрудника.

Задание 2. [7]

Написать запрос, выводящий список всех job ID из таблицы сотрудников без повторений.

Задание 3. [8]

Написать запрос, выводящий: код сотрудника (с псевдонимом Emp #), фамилию (Employee), JOB_ID (Job), дату приёма на работу (Hire Date).

Задание 4. [9]

Написать запрос, выводящий список (в один столбец) следующих данных: фамилия и JOB_ID (через запятую). Назвать столбец “Employee and Title”.

Фильтрация и сортировка. [III]

Замечание 1. Псевдонимы не могут использоваться в условных выражениях WHERE.

Замечание 2. Оператор BETWEEN эквивалентен двум операторам сравнения (field_1 >= x1 AND field_1 <= x2).

Замечание 3. Оператор IN эквивалентен нескольким операторам = (field_1 = x1 OR field_1=x2…).

Замечание 4. Подстановочные символы в операторе LIKE. Если необходимо использовать сами подстановочные символы (_ и %), можно использовать оператор ESCAPE: field_1 LIKE ‘%SA_%’ ESCAPE ‘\’;

Замечание 5. Без использования ORDER BY результат запроса не детерминирован. В ORDER BY можно использовать псевдонимы столбцов, выражения либо порядковые номера столбцов.

Задание 1. [1]

Написать запрос, выводящий для сотрудников, зарабатывающих больше 12000, следующие поля: Фамилия, оклад.

Задание 2. [3]

Написать запрос, выводящий поля Фамилия, оклад для сотрудников, оклад которых находится не в интервале от 5 000..12 000.

Задание 3. [4]

Написать запрос, выводящий поля Фамилия, job_ID и дата приема на работу для сотрудников с фамилиями ‘Matos’ и ‘Taylor’. Результат упорядочить по возрастанию даты приема на работу.

Задание 4. [6]

Написать запрос, выводящий поля Фамилия (псевдоним ‘Employee’) и оклад (псевдоним ‘Monthly Salary’) сотрудников, зарабатывающих в диапазоне 5 000..12 000 и работающих в отделах с ID 20 или 50.

Задание 5. [8]

Написать запрос, выводящий поля Фамилия и job_ID сотрудников, у которых нет менеджера.

Задание 6. [10]

Написать запрос, выводящий поля Фамилия и оклад сотрудников, которые зарабатывают больше некоторого указанного (по запросу) количества.

Задание 7. [13]

Написать запрос, выводящий фамилии сотрудников, которые содержат букву ‘a’ и букву ‘e’.

Задание 8. [100]

Написать запрос, выводящий фамилию, оклад сотрудников, у которых job_ID начинается с ‘AC_’.

Функции одной строки. [IV]

Замечание 1. SYSDATE – системное время сервера, к которому выполнено подключение. Если нужно актуальное время сессии, используется другая функция (CURRENT_DATE).

Задание 1. [1]

Вывести системную дату, обозначить столбец Date.

Задание 2. [6]

Для каждого сотрудника вывести: фамилию, количество проработанных на данный момент месяцев (столбец обозначить MONTHS_WORKED). Количество месяцев округлить до целого. Упорядочить результат по количеству отработанных месяцев.

Задание 3. [8]

Вывести первые 8 символов фамилии и через пробел количество звездочек, соответствующее уровню оклада. Одна звездочка – 1000 (долларов). Обозначить столбец EMPLOYEES_AND_THEIR_SALARIES. Упорядочить по убыванию уровня з.п.

Функции преобразования и условные выражения. [V]

Замечание 1. Функция Nvl – преобразует значения NULL первого аргумента в значение второго аргумента (если значение первого аргумента не равно NULL, то возвращается оно, без изменений), Nlvl2 – если expr1 не равно NULL, то возвращает expr2, иначе expr3, Nullif – если выражения равны, то NULL, иначе первое выражение, Coalesce – возвращает первое выражение из списка аргументов, не равное NULL.

Задание 1. [3]

Фамилия, дата приема на работу и соответствующий день недели. Обозначить столбец с днем недели как DAY. Упорядочить результат по дням недели, начиная со вторника. (вторник, среда… понедельник).

Задание 2. [4]

Фамилия и размер комиссии (commission_pct). Если комиссии нет, вывести слово ‘No Commission’. Обозначить столбец с размером комиссии ‘COMM’.

Задание 3. [5]

Вывести фамилию сотрудника и поле Grade. Grade формируется на основе JOB_ID по следующей таблице соответствий:

AD_PRES A

ST_MAN B

IT_PROG C

SA_REP D

ST_CRERK E

None of the above 0

Использовать функцию DECODE.

Задание 4. [6]

Аналогично 3, но использовать CASE

Групповые (агрегирующие) функции. [VI]

Замечание 1. Перед аргументом функции можно указывать DISTINCT либо ALL (по умолчанию). Все групповые функции игнорируют значения NULL: если необходимо их учитывать, надо использовать функции работы с неопределенными значениями (NVL и т.п.). Название столбца в аргументе функции может быть заменено выражением.

Замечание 2. Функции AVG, SUM, VARIANCE, STDDEV – оперируют только с числовыми типами аргументов. Функции MIN, MAX – с числовыми, строковыми и типами дата.

Замечание 3. Функция COUNT имеет три формата:

COUNT(*) – количество строк в запросе, удовлетворяющих запросу, включая повторяющиеся и строки с пустыми значениями;

COUNT(expr) – количество строк с непустыми значениями для expr;

COUNT(DISTINCT expr) – количество уникальных строк с непустыми значениями для expr.

Замечание 4. Если использовать функции группировки без GROUP BY, вся таблица будет трактоваться как одна большая группа. Иногда необходимо разбить таблицу на несколько групп – в этом случае используется GROUP BY.

Замечание 5. Если вы используете группировку, то нельзя в списке того, что надо выбрать, указывать отдельный столбец, если его нет в критериях группировки. Другими словами, все столбцы выборки должны быть включены в критерии группировки.

Замечание 6. С помощью условия в WHERE исключаются некоторые строки перед разбиением на группы. Если необходимо ограничить (отфильтровать) строки на основе значений групповой функции, необходимо использовать GROUP BY и HAVING.

Замечание 7. Псевдонимы столбцов не могут быть использованы в GROUP BY (как и в WHERE).

Замечание 8. В ORDER BY можно использовать групповые функции.

Замечание 9. Групповые функции могут быть вложенными с глубиной вложенности 2. GROUP BY обязателен, когда используются вложенные групповые функции.

Задание 1. [4]

Вывести максимальный, минимальный оклады, сумму всех окладов сотрудников и средний оклад сотрудников. Столбцы назвать Maximum, Minimum, Sum и Average. Округлить результаты до ближайшего целого.

Задание 2. [5]

Аналогично 1, но вывести дополнительно поле job_id и результаты по каждому коду работы.

Задание 3. [6]

Вывести job_id и количество сотрудников с такой работой.

Модифицировать запрос, чтобы результат был для job_id по запросу (например, IT_PROG).

Задание 4. [7]

Вывести общее количество менеджеров без их перечисления.

Задание 5. [8]

Вывести разницу между максимальным и минимальным окладами.

Задание 6. [9]

Вывести ID менеджера, оклад самого низкооплачиваемого сотрудника этого менеджера. Исключить сотрудников с неопределенным менеджером. Исключить группы с минимальным окладом 6000 или менее. Сортировать по убыванию минимального оклада.

Задание 7. [10]

Вывести строку с: общим количеством сотрудников, а также с детализацией количества сотрудников, работающих с 1995, 1996, 1997 и 1998 гг. Обозначить столбцы соответственно.

Задание 8. [11]

Вывести job_id и сумму окладов для этой работы с детализацией по отделам: 20 50 80 90 и всего.

Обозначить столбцы Dept 20, Dept 50, Dept 80, Dept 90, Total.

Объединение таблиц. [VII]

Замечание 1. NATURAL JOIN – объединение на основе равенства значений столбцов с одинаковыми именами. Если у нескольких полей одинаковые названия, но разные типы, будет ошибка. USING – на основе равенства значений определенного столбца. ON – на основе произвольного условия. Все это внутренние (INNER) объединения.

LEFT/RIGHT/FULL OUTER – для внешних объединений.

CROSS JOIN – для декартова произведения. Декартово произведение можно получить и выборкой столбцов из разных таблиц без условий связей. Однако, использование явного декартова произведения повышает читабельность SQL-запроса.

Замечание 2. Если есть столбцы с одинаковыми именами в двух таблицах, но объединение идет не по ним, для их использования обязательно надо уточнять таблицу. Если вы начали использовать псевдоним, то его и надо использовать во всем запросе. Иначе будет ошибка.

Замечание 3. Объединение происходит слева-направо в порядке упоминания объединений.

Задание 1. [1]

Вывести полные адреса отделов: location_id, street_address, city, state_province, country_name. Использовать NATURAL JOIN.

Задание 2. [2]

Для всех сотрудников вывести: фамилию, Id отдела, название отдела. Использовать USING.

Задание 3. [3]

Вывести фамилию, job_id, department_id, department_name всех сотрудников, работающих в Toronto.

Задание 4. [4]

Вывести фамилию сотрудника (псевдоним “Employee”), Id сотрудника (“Emp #”), фамилию менеджера (“Manager”) и Id менеджера (“Mgr #”).

Задание 5. [5]

Аналогично 4, но еще выводить сотрудников без менеджеров. Упорядочить результат по Id сотрудника.

Задание 6. [6]

Коллеги. Вывести Id отдела, фамилию сотрудника, фамилию коллеги. Коллега – сотрудник, работающий в том же отделе. Упорядочить по Id отдела, фамилии сотрудника и фамилии коллеги.

Задание 7. [7]

Вывести фамилию, Job_id, название отдела, оклад, Grade_level (необходимо использовать объединение трех таблиц).

Задание 8. [8]

Вывести информацию о всех сотрудников, поступивших на работу после ‘Davies’. Информация включает имя сотрудника и дату приема на работу.

Задание 9. [9]

Вывести информацию о всех сотрудников, кто пришел работать до начала работы своих менеджеров. Информация включает фамилию сотрудника, дату приема на работу сотрудника, фамилию менеджера, дату приема на работу менеджера.

Subqueries. [VIII]

Замечание 1. Подзапросы могут участвовать в выражениях WHERE, HAVING, FROM

Замечание 2. operator для подзапроса бывает двух видов: однострочный (>, =, >=, <, <=, <>) и многострочный (IN, ANY, ALL, EXISTS).

Замечание 3. У ANY есть синоним – SOME

=ANY эквивалентно IN

<>ALL эквивалентен NOT IN

Задание 1. [1]

Отобразить окно для ввода фамилии. Вывести фамилии и даты приема на работу сотрудников, работающих в том же отделе, что с сотрудник с введенной фамилией.

Задание 2. [2]

Вывести id сотрудника, фамилию и оклад – для сотрудников, зарабатывающих выше среднего уровня.

Задание 3. [3]

Вывести id сотрудника и фамилию – для сотрудников, работающих в тех же отделах, что и сотрудники, фамилия которых содержит букву ‘u’.

Задание 4. [4]

Вывести фамилию, id отдела, job_id для сотрудников, работающих в отделах, у которых location_id = 1700. Модифицировать, чтобы location_id можно было вводить.

Задание 5. [5]

Вывести фамилию и оклад сотрудников, работающих под руководством сотрудника ‘King’.

Задание 6. [6]

Вывести Department_id, фамилию, job_id для сотрудников, работающих в отделах с названием ‘Executive’.

Задание 7. [7]

Вывести фамилии сотрудников, оклад которых выше любого из сотрудников из отдела с id = 60.

Задание 8. [8]

Вывести id сотрудника, фамилию и оклад – для сотрудников, зарабатывающих выше среднего и работающих в тех же отделах, что и сотрудники, фамилия которых содержит букву ‘u’.

Множественные операции. [IX]

Замечание 1. Операторы множеств имеют одинаковый приоритет и выполняются слева направо (управлять можно с помощью скобок).

Замечание 2. Типы не обязательно должны в точности совпадать. Важно, чтобы столбцы из второго запроса относились к той же группе типа, что и столбцы из первого запроса.

Замечание 3. Дубли автоматически исключаются, кроме использования оператора UNION ALL.

Задание 1. [1]

Вывести список ID отделов, в которых нет job_id = ST_CLERK (job_id из таблицы employees). Для получения результата использовать операторы работы с множествами.

Задание 2. [2]

Вывести список стран (ID, название), в которых не расположен ни один отдел. Для получения результата использовать операторы работы с множествами.

Задание 3. [3]

Вывести список работ для отделов 10, 50, 20 (именно в таком порядке). Отобразить job ID, department ID. Использовать операторы работы с множествами.

Задание 4. [4]

Вывести ID сотрудника и job ID для сотрудников, кто работает на работе не в первый раз (т.е. работали раньше на этой работе и вернулись опять).

Задание 5. [5]

Составить запрос, выводящий фамилию, код отдела всех сотрудников (из таблицы employees). В том числе для сотрудников, у которых не определен отдел.

Составить запрос, выводящий код отдела, название отдела всех отделов (из таблицы departments). В том числе отделов, в которых не работает ни одного сотрудника.

Объединить результаты с двумя запросами в один с помощью множественных операций.

Работа с DDL. [XI]

Замечание 1. Схема – это коллекция объектов. У схемы есть владелец, название схемы совпадает с его именем. У каждого пользователя только одна схема. Для доступа к объектам, которые не принадлежат пользователю, необходимо использовать префиксы соответствующей схемы.

Замечание 2. При создании таблицы можно определить значение по умолчанию для столбца с помощью ключевого слова DEFAULT. С помощью этой опции при вставке строки без указания значения для некоторого столбца можно обеспечить вставку значения по умолчанию и избежать вставки значения NULL.

Замечание 3. Ограничения целостности уровня таблицы определяются в конце определения таблицы и должны включать ссылки на столбцы, для которых они действуют, в круглых скобках. Функционально ограничения на столбцы и таблицу одинаковы.

Ограничение NOT NULL должно определяться на уровне столбца.

Ограничения, которые работают с несколькими столбцами, должны определяться на уровне таблицы.

Замечание 4. Ограничения могут быть созданы на этапе создания таблицы или добавлены позже. Они могут быть также временно отключены.

Замечание 5. Ограничение UNIQUE допускает пустые значения. Кроме того, пустые значения автоматически удовлетворяют этому ограничению (может содержаться несколько пустых значений). Использование UNIQUE на уровне таблицы позволяет включить в ограничение несколько столбцов.

Замечание 6. У таблицы может быть только один первичный ключ. Первичный ключ – это столбец или набор столбцов, которые однозначно идентифицируют строки в таблице. Это ограничение накладывает ограничение уникальности на столбец или набор столбцов и исключает наличие пустых значений.

Замечание 7. Внешний ключ должен соответствовать значению главной таблицы или быть неопределенным. Внешние ключи могут определяться как на уровне столбцов, так и на уровне таблицы. По умолчанию строка в главной таблице, на которую ссылаются, не может быть удалена.

Задание 1. [4]

Создать таблицу Employees2 на основе структуры таблицы Employees. Включить только столбцы Employee_id, First_name, last_name, salary, department_id. Назовите соответствующие столбцы в новой таблице: ID, FIRST_NAME, LAST_NAME, SALARY, DEPT_ID.

Задание 2. [5]

Измените таблицу Employees2 и переведите ее в режим только для чтения.

Задание 3. [6]

Добавьте строку (34, Grant, Marcie, 5678, 10).

Задание 4. [7]

Переведите таблицу Employees2 в режим чтения/записи. Добавьте строку из задания 3 повторно.

Задание 5. [8]

Удалите таблицу Employees2.

Работа с некоторыми другими объектами [XII]

Замечание 1. С помощью представлений можно создавать автоматически поддерживаемые выборки данных (логические таблицы) из одной или нескольких таблиц или других представлений.

Таблицы, на которых основано представление, называются базовыми таблицами. В основе представления лежит оператор SELECT (этот оператор сохраняется в словаре данных).

Использование ключевого слова FORCE позволяет создать представление, даже если базовые таблицы не существуют (по умолчанию NOFORCE).

Ключевое слово WITH CHECK OPTION указывает на то, что строки в базовых таблицах, изменяемые и вставляемые через представление, должны соответствовать критерию отбора в запросе, определяющем представление.

Замечание 2. С помощью последовательностей можно генерировать значения для первичных ключей. Последовательности не привязаны к конкретной таблице.

Замечание 3. С помощью индексов можно повышать производительность выборки данных из таблиц. Индексы нельзя изменять. Для изменения надо удалить и создать заново.

Замечание 4. С помощью синонимов можно создавать альтернативные имена для объектов.

PUBLIC – синоним доступен для всех пользователей. Для доступа к объектам из других схем надо использовать префиксы с названием схемы. Более короткий вариант – использование синонимов.

Часть I.

Задание 1. [1]

Сотрудники отдела HR хотят скрыть часть информации из таблицы Employees. Создайте представление Employees_VU (столбцы ‘Employee_id’, ‘Employee’, ‘Department_id’) на основе таблицы Employees (поля ID, last_name, номер отдела).

Задание 2. [2]

Вывести все строки из полученного в задании 1 представления.

Задание 3. [3]

Вывести employee, employee_id из представления.

Задание 4. [4]

Отделу 50 нужна информация о сотрудниках. Создайте представление с именем DEPT50, которое содержит ID сотрудника, last_name и ID отдела для сотрудников, работающих в отделе 50. Для защиты запретите переименовать ID отдела.

Задание 5. [5]

Отобразите структура и все строки из DEPT50.

Задание 6. [6]

Попробуйте изменить номер отдела для сотрудника с employee = ‘Matos’ на 80.

Часть II

Задание 7. [7]

Необходима последовательность для использования в качестве первичного ключа в таблице DEPT. Она начинается с 200 и имеет максимум 1000, приращение = 10. Назовите последовательность DEPT_ID_SEQ.

Задание 8. [8]

Протестируйте последовательность. Создайте новый отдел ‘Education’, ID отдела получите с помощью последовательности. Создайте отдел ‘Administration’, ID отдела получите с помощью последовательности.

Задание 9. [9]

Создайте nonunique индекс для столбца Name в таблице DEPT.

Задание 10. [10]

Создайте синоним для таблицы EMPLOYEES под названием EMP.