**Лекция 21**

**1.Внешнее объединение**

Часто бывает полезна операция объединения двух запросов,

в которой второй запрос выбирает строки, исключенные первым. Такая операция называется внешним объединением.

Рассмотрим пример.

Пусть в таблице STUDENT имеются записи о студентах, в которых не указан идентификатор университета.

Требуется составить список студентов с указанием наименований университетов для тех студентов, у которых эти данные есть, не исключая тех, у которых UNIV\_ID не указан.

Формируем объединение из двух запросов, один из которых выполняет выборку студентов с названиями UNIV\_ID , а второй с NULL-значениями в поле UNIV\_ID.

SELECT SURNAME, NAME, UNIV\_NAME

FROM STUDENT, UNIVERSITY

WHERE STUDENT.UNIV\_ID=UNIVERSITY.UNIV\_ID

UNION

SELECT SURNAME, NAME, ‘Неизвестен’

FROM STUDENT

WHERE UNIV\_ID IS NULL

ORDER BY 1;

Для совместимости столбцов объединяемых запросов константу ‘Неизвестен’ во втором запросе следует дополнить пробелами так, чтобы ее длина соответствовала длине поля UNIV\_NAME, или использовать функцию CAST. В некоторых СУБД согласование типов поля и замещающей его текстовой константы осуществляется автоматически.

**2.Соединение таблиц с использованием JOIN**

Если в операторе SELECT после ключевого слова FROM указывается не одна, а две таблицы, то в результате выполнения запроса, в котором отсутствует предложение WHERE каждая строка одной таблицы будет соединена с каждой строкой второй.

Такая операция называется *декартовым произведением или полным (CROSS) соединением таблиц.*

В случае использования этой операции в ответе на запрос количество записей будет равно произведению числа записей в соединяемых таблицах, т.е. может оказаться чрезвычайно большим.

Соединение таблиц имеет смысл в том случае, если соединяются не все строки исходных таблиц, а только интересующие пользователя.

Ограничение осуществляется предложением WHERE. Таким образом SQL позволяет выводить информацию из нескольких таблиц, связывая их по значениям определенных полей.

Например, если необходимо получить фамилии студентов и для каждого студента названия университетов, расположенных в городе, где живет студент:

SELECT STUDENT.SURNAME, UNIVERSITY.UNIV\_NAME, STUDENT.CITY

FROM STUDENT, UNIVERSITY

WHERE STUDENT.CITY=UNIVERSITY.CITY;

Соединение, использующее предикаты, основанные на равенствах, называется *эквисоединением*.

Рассмотренный пример соединения таблиц относится к виду так называемого внутреннего (INNER) соединения. При этом соединяются только те строки таблиц, для которых истинным является предикат, задаваемый в предложении ON выполняемого запроса.

Приведенный выше запрос может быть записан иначе, с использованием ключевого слова JOIN.

SELECT STUDENT.SURNAME, UNIVERSITY.UNIV\_NAME,

STUDENT.CITY

FROM STUDENT INNER JOIN UNIVERSITY

ON STUDENT.CITY = UNIVERSITY.CITY;

Ключевое слово INNER в запросе может быть опущено, так как эта опция в операторе JOIN действует по умолчанию.

Рассмотренный выше случай полного соединения (декартова произведения) таблиц с использованием ключевого слова JOIN будет выглядеть следующим образом:

SELECT \* FROM STUDENT JOIN UNIVERSITY;

что эквивалентно

SELECT \* FROM STUDENT, UNIVERSITY;

Заметим, что в СУБД Oracle задаваемый стандартом языка SQL оператор JOIN не поддерживается.

**3.Операции соединения таблиц посредством ссылочной целостности**

Информация в таблицах STUDENT и EXAM\_MARKS уже связана посредством поля STUDENT\_ID. В таблице STUDENT поле STUDENT\_ID является первичным ключом, а в таблице EXAM\_MARKS — ссылающимся на него внешним ключом.

Состояние связанных таким образом таблиц называется *состоянием ссылочной целостности.*

В данном случае ссылочная целостность этих таблиц подразумевает, что каждому значению поля STUDENT\_ID в таблице EXAM\_MARKS обязательно соответствует такое же значение поля STUDENT ID в таблице STUDENT.

Другими словами, в таблице EXAM\_MARKS не может быть записей, имеющих идентификаторы студентов, которых нет в таблице STUDENT. Стандартное применение операции соединения состоит в извлечении данных в терминах этой связи.

Чтобы получить список фамилий студентов с полученными ими оценками и идентификаторами предметов, можно использовать следующий запрос:

SELECT SURNAME, MARK, SUBJ\_ID

FROM STUDENT, EXAM\_MARKS

WHERE STUDENT.STUDENT\_ID = EXAM\_MARKS.STUDENT\_ID;

Тот же результат может быть получен при использовании в запросе для задания операции соединения таблиц ключевого слова JOIN. Запрос с оператором JOIN выглядит следующим образом:

SELECT SURNAME, MARK

FROM STUDENT JOIN EXAM\_MARKS

ON STUDENT.STUDENT\_ID = EXAM\_MARKS.STUDENT\_ID;

Хотя выше речь шла о соединении двух таблиц, можно сформировать запросы путем соединения более чем двух таблиц.

Пусть требуется найти фамилии всех студентов, получивших неудовлетворительную оценку, вместе с названиями предметов обучения, по которым получена эта оценка.

SELECT SUBJ\_NAME, SURNAME, MARK

FROM STUDENT, SUBJECT, EXAM\_MARKS

WHERE STUDENT.STUDENT\_ID = EXAM\_MARKS.STUDENT\_ID

AND SUBJECT.SUBJ\_ID = EXAM\_MARKS.SUBJ\_ID

AND EXAM\_MARKS.MARK = 2;

Toже самое с использованием оператора JOIN:

SELECT SUBJ\_NAME, SURNAME, MARK

FROM STUDENT JOIN SUBJECT JOIN EXAM\_MARKS

ON STUDENT.STUDENT\_ID = EXAM\_MARKS.STUDENT\_ID

AND SUBJECT.SUBJ\_ID = EXAM\_MARKS.SUBJ\_ID

AND EXAM\_MARKS.MARK = 2;

Как отмечалось ранее, при использовании внутреннего (INNER) соединения таблиц соединяются только те их строки, в которых совпадают значения полей, задаваемые в запросе предложением WHERE.

Однако во многих случаях это может привести к нежелательной потере информации. Рассмотрим еще раз приведенный выше пример запроса на выборку списка фамилий студентов с полученными ими оценками и идентификаторами предметов. При использовании, как это было сделано в рассматриваемом примере, внутреннего соединения в результат запроса не попадут студенты, которые еще не сдавали экзамены, и которые, следовательно, отсутствуют в таблице EXAM\_MARKS.

Если же необходимо иметь записи об этих студентах в выдаваемом запросом списке, то можно присоединить сведения о студентах, не сдававших экзамен, путем использования оператора UNION с соответствующим запросом. Например, следующим образом:

SELECT SURNAME, CAST MARK AS CHAR(1), CAST SUBJ\_ID AS CHAR (10)

FROM STUDENT, EXAM\_MARKS

WHERE STUDENT.STUDENT\_ID = EXAM\_MARKS.STUDENT\_ID

UNION

SELECT SURNAME, CAST NULL AS CHAR (1), CAST NULL AS CHAR (10)

FROM STUDENT

WHERE NOT EXISTS

(SELECT \*

FROM EXAM\_MARKS

WHERE STUDENT.STUDENT\_ID = EXAM\_MARKS.STUDENT\_ID);

(здесь функция преобразования типов CAST используется для обеспечения совместимости типов полей объединяемых запросов).

Нужный результат может быть получен и путем использования внешнего соединения, точнее, одной из его разновидностей —левого внешнего соединения, с применением которого запрос будет выглядеть следующим образом:

SELECT SURNAME, MARK

FROM STUDENT LEFT OUTER JOIN EXAM\_MARKS

ON STUDENT.STUDENT\_ID = EXAM\_

MARKS.STUDENT ID;

При использовании левого соединения расширение выводимой таблицы осуществляется за счет записей входной таблицы, имя которой указано слева от оператора JOIN.

Приведенный выше запрос может быть реализован и с применением правого внешнего соединения. Он будет иметь следующий вид:

SELECT SURNAME, MARK

FROM EXAM\_MARKS RIGHT OUTER JOIN STUDENT

ON EXAM\_MARKS.STUDENT\_ID = STUDENT.STUDENT\_ID;

Здесь таблица STUDENT, за счет записей которой осуществляется расширение выводимой таблицы, указана справа от оператора JOIN.

Иногда возникает необходимость включения в результат запроса записей из обеих (правой и левой) соединяемых таблиц, для которых не удовлетворяется условие соединения. Такое соединение называется *полным внешним соединением* и осуществляется указанием в запросе ключевых слов FULL OUTER JOIN или UNION JOIN.

**3.Использование псевдонимов при соединении таблиц**

Часто при запросе информации необходимо осуществлять соединение таблицы с ее же копией.

Например, это требуется в случае, когда нужно найти фамилии студентов, имеющих одинаковые имена. При соединении таблицы с ее же копией вводят псевдонимы (алиасы) таблицы. Запрос для поиска фамилий студентов, имеющих одинаковые имена, выглядит следующим

образом:

SELECT FIRST.SURNAME, SECOND.NAME

FROM STUDENT FIRST, STUDENT SECOND

WHERE FIRST.NAME = SECOND.NAME;

В этом запросе введены два псевдонима для одной таблицы STUDENT, что позволяет корректно задать выражение, связывающее две копии таблицы.

Чтобы исключить повторения строк в выводимом результате запроса из-за повторного сравнения одной и той же пары студентов, необходимо задать порядок следования для двух значений так, чтобы одно значение было меньше, чем другое, что делает предикат асимметричным.

SELECT FIRST.SURNAME, SECOND.NAME

FROM STUDENT FIRST, STUDENT SECOND

WHERE FIRST.NAME = SECOND.NAME

AND FIRST.SURNAME < SECOND.SURNAME;