**Лабораторная работа № 5.** **Создание многотабличных запросов.**

Цель работы: Получить правктические навыки формирования запросов с объединением таблиц

***Краткие теоретические сведения***

1. **Неявное объединение таблиц**

Информация о предметной области в базе данных распределена по нескольким таблицам. Часто бывает необходимо для получения полной информации о каком-либо объекте объединять таблицы. Например, для получения информации о названиях и датах продажи товаров необходимо объединить таблицы sales («Продажи»), goods («Товары») и ledger («Журнал учета продаж»). Для объединения таблиц можно использовать команду SELECT.

Запрос на объединение таблиц в этом случае будет иметь вид, показанный в примере 5.1

Пример 5.1. Запрос на объединение таблиц.

**USE shop;**

**SELECT \* From sales S, goods G**

**WHERE S.goodId=G.goodId**

В этом запросе сначала происходит объединение всех строк таблиц sales и goods. Так как в используемом примере базы данных таблицы sales и goods содержат по 8 записей, то в результате их объединения будет сформирована таблица, содержащая 64 записиа (декартово произведение двух групп). Затем предложение WHERE отберет только те записи, в которых значения goodId совпадают. В результате будет получена следующая таблица.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **recordId** | **goodId** | **quantity** | **price** | **saleId** | **goodId** | **good** | **unitId** | **groupId** |
| 1 | 1 | 3.00 | 3.00 | 1 | 1 | мука | 1 | 1 |
| 2 | 3 | 2.00 | 8.50 | 1 | 3 | свинина | 1 | 2 |
| 3 | 5 | 2.50 | 5.50 | 1 | 5 | треска | 1 | 3 |
| 4 | 1 | 2.00 | 3.00 | 2 | 1 | мука | 1 | 1 |
| 5 | 6 | 1.50 | 4.50 | 2 | 6 | минтай | 1 | 3 |
| 6 | 7 | 1.00 | 2.00 | 3 | 7 | квас | 3 | 4 |
| 7 | 4 | 2.00 | 14.50 | 3 | 4 | говядина | 1 | 2 |
| 8 | 3 | 1.00 | 1.00 | 3 | 3 | свинина | 1 | 2 |

Для объединения трех и более таблиц нужно в запрос 5.1 нужно добавить имена объединяемых таблиц и условия отбра строк.

1. **Вертикальное объединение таблиц**

Язык Transact-SQL поддерживает три оператора вертикального объединения:

* UNION;
* INTERSECT;
* EXCEPT.

Эти операторы обычно применяются для объединения таблиц, полученных в результате запросов.

**Оператор UNION**

Оператор UNION объединяет результаты двух или более запросов в один результирующий набор, в который входят все строки, принадлежащие всем запросам в объединении.

Фактически оператор UNION производит вертикальное объединение таблиц, являющихся результатом двух или более запросов. Соответственно, результатом объединения двух таблиц является новая таблица, содержащая все строки, входящие в одну из исходных таблиц или в обе эти таблицы.

Общая форма оператора UNION выглядит таким образом:

**select\_1 UNION [ALL] select\_2 {[UNION [ALL] select\_3]}...**

Параметры select\_1, select\_2, ... представляют собой инструкции SELECT, которые создают объединение. Если используется параметр ALL, отображаются все строки, включая дубликаты. В операторе UNION параметр ALL имеет то же самое значение, что и в списке выбора SELECT, но с одним отличием: для списка выбора SELECT этот параметр применяется по умолчанию, а для оператора UNION его нужно указывать явно.

В примере 5.2 показан запрос для создания вертикального объединения результирующих наборов двух запросов к одной и той же таблице Sales с разными условиями отбора записей.

Пример 5.2. Объединение результатов запросов с помощью инструкции UNION

**USE Shop;**

**SELECT \* FROM Sales WHERE saleId=1**

**UNION ALL**

**SELECT \* FROM Sales WHERE price>5 ;**

Результат выполнения этого запроса:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **recordId** | **goodId** | **quantity** | **price** | **saleId** |
| 1 | 1 | 3.00 | 3.00 | 1 |
| 2 | 3 | 2.00 | 8.50 | 1 |
| 3 | 5 | 2.50 | 5.50 | 1 |
| 2 | 3 | 2.00 | 8.50 | 1 |
| 3 | 5 | 2.50 | 5.50 | 1 |
| 7 | 4 | 2.00 | 14.50 | 3 |

Несмотря на то, что в таблице «Продажи» (Sales) имеется всего три записи с параметром saleId=1, в результирующем наборе запроса 5.2 таких записей 5. Поскольку после слова UNION указан параметр ALL, в результирующий набор попали повторяющиеся записи, отобранные как первым, так и вторым запросом. Если тот же запрос использовать без слова ALL как показано в примере 5.3, то в результирующем наборе будет только 3 записи.

Пример 5.3. Объединение результатов запросов с помощью инструкции UNION

**USE Shop;**

**SELECT \* FROM Sales WHERE saleid=1**

**UNION**

**SELECT \* FROM Sales WHERE price>5 ;**

Результат выполнения этого запроса:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **recordId** | **goodId** | **quantity** | **price** | **saleId** |
| 1 | 1 | 3.00 | 3.00 | 1 |
| 2 | 3 | 2.00 | 8.50 | 1 |
| 3 | 5 | 2.50 | 5.50 | 1 |
| 7 | 4 | 2.00 | 14.50 | 3 |

Объединяемые с помощью инструкции UNION результирующие наборы (таблицы) должны содержать одинаковое число столбцов, а соответствующие столбцы должны иметь совместимые типы данных.

Результат объединения можно упорядочить, используя предложение ORDER BY в последней инструкции SELECT, как это показано в примере 5.4. Предложения GROUP BY и HAVING можно применять с отдельными инструкциями SELECT, но не в самом объединении.

Пример 5.4. Упорядочивание результатов объединения двух таблиц

**USE Shop;**

**SELECT \* FROM Sales WHERE saleid=1**

**UNION**

**SELECT \* FROM Sales WHERE price>5**

**ORDER BY price;**

Результат выполнения этого запроса:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **recordId** | **goodId** | **quantity** | **price** | **saleId** |
| 1 | 1 | 3.00 | 3.00 | 1 |
| 3 | 5 | 2.50 | 5.50 | 1 |
| 2 | 3 | 2.00 | 8.50 | 1 |
| 7 | 4 | 2.00 | 14.50 | 3 |

Вместо оператора UNION можно применить оператор OR, если все инструкции SELECT,

соединенные одним или несколькими операторами UNION, ссылаются на одну и ту же таблицу.

**Операторы INTERSECT и EXCEPT**

Два других оператора для работы с результирующими наборами, INTERSECT и EXCEPT, определяют пересечение и разность соответственно. Под пересечением понимается набор строк, которые принадлежат обеим таблицам. А разность двух таблиц определяется как все значения, которые принадлежат к первой таблице и не присутствуют во второй. В примере 5.5 показано использование оператора INTERSECT.

Пример 5.5. Применение оператора INTERSECT

**USE Shop;**

**SELECT \* FROM Sales WHERE saleid=1**

**INTERSECT**

**SELECT \* FROM Sales WHERE price>5;**

Результат выполнения этого запроса:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **recordId** | **goodId** | **quantity** | **price** | **saleId** |
| 2 | 3 | 2.00 | 8.50 | 1 |
| 3 | 5 | 2.50 | 5.50 | 1 |

Использование параметра ALL ни с оператором INTERSECT, ни с оператором EXCEPT в языке Transact-SQL не поддерживается. Использование оператора EXCEPT показано в примере 5.6.

Пример 5.6. Применение оператора EXCEPT

**USE Shop;**

**SELECT \* FROM Sales WHERE saleid=1**

**EXCEPT**

**SELECT \* FROM Sales WHERE price>5;**

Результат выполнения этого запроса:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **recordId** | **goodId** | **quantity** | **price** | **saleId** |
| 1 | 1 | 3.00 | 3.00 | 1 |

1. **Горизонтальное объединение таблиц**

Для горизонтального объединения таблиц в языке Transact-SQL используется оператор объединения JOIN, который позволяет извлекать данные более чем из одной таблицы.

Существует пять типов объединения:

1. **JOIN** – левая\_таблица JOIN правая\_таблица ON условия\_объединения
2. **LEFT JOIN** – левая\_таблица LEFT JOIN правая\_таблица ON условия\_объединения
3. **RIGHT JOIN** – левая\_таблица RIGHT JOIN правая\_таблица ON условия\_объединения
4. **FULL JOIN** – левая\_таблица FULL JOIN правая\_таблица ON условия\_объединения
5. **CROSS JOIN** – левая\_таблица CROSS JOIN правая\_таблица

|  |  |  |
| --- | --- | --- |
| **Краткий синтаксис** | **Полный синтаксис** | **Описание** |
| JOIN | INNER JOIN | Из строк левой\_таблицы и правой\_таблицы объединяются и возвращаются только те строки, по которым выполняются условия\_объединения. |
| LEFT JOIN | LEFT OUTER JOIN | Возвращаются все строки левой\_таблицы (ключевое слово LEFT). Данными правой\_таблицы дополняются только те строки левой\_таблицы, для которых выполняются условия\_объединения. Для недостающих данных вместо строк правой\_таблицы вставляются NULL-значения. |
| RIGHT JOIN | RIGHT OUTER JOIN | Возвращаются все строки правой\_таблицы (ключевое слово RIGHT). Данными левой\_таблицы дополняются только те строки правой\_таблицы, для которых выполняются условия\_объединения. Для недостающих данных вместо строк левой\_таблицы вставляются NULL-значения. |
| FULL JOIN | FULL OUTER JOIN | Возвращаются все строки левой\_таблицы и правой\_таблицы. Если для строк левой\_таблицы и правой\_таблицы выполняются условия\_объединения, то они объединяются в одну строку. Для строк, для которых не выполняются условия\_объединения, NULL-значения вставляются на место левой\_таблицы, либо на место правой\_таблицы, в зависимости от того данных какой таблицы в строке не имеется. |
| CROSS JOIN | - | Объединение каждой строки левой\_таблицы со всеми строками правой\_таблицы. Этот вид объединения иногда называют декартовым произведением. |

Особенности различных типов объединений удобно рассмотреть на примерах работы с частью базы данных, в которой хранится информация о сотрудниках магазина. Диаграмма этой части базы данных представлена на рис. 5.1

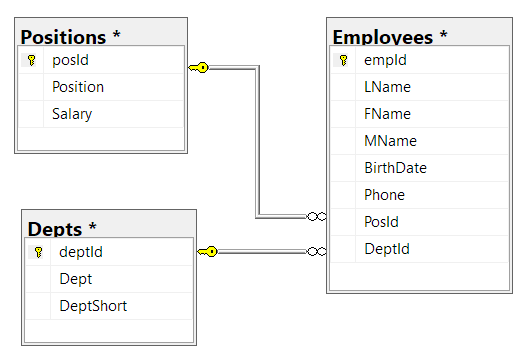


Рис.5.1 Диаграмма части базы данных, в которой хранится информация о сотрудниках.

Содержимое таблиц Positions (Должности), Depts(Отделы) и Employees (Cотрудники) приведено в таблицах 5.1-5.3

Таблица 5.1. Заполненная таблица Depts.

|  |  |
| --- | --- |
| **deptId** | **Dept** |
| 1 | Дирекция |
| 2 | Бухгалтерия |
| 3 | Отдел продаж |
| 4 | Склад |
| 5 | Транспортный отдел |

Таблица 5.2. Заполненная таблица Positions.

|  |  |
| --- | --- |
| **posId** | **position** |
| 1 | Директор |
| 2 | Главный бухгалтерия |
| 3 | Бухгалтер |
| 4 | Продавец |
| 5 | Грузчик |
| 6 | Водитель |

Таблица 5.3. Заполненная таблица Employees

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **emp Id** | **LName** | **FName** | **MName** | **BirthDate** | **Phone** | **deptId** | **posId** |
| 1000 | Сытый | Анатолий | Александрович | 30.08.1957 | 80291122345 | 1 | 1 |
| 1001 | Сизов | Александр | Сергеевич | 12.12.1983 | 80335643212 | 3 | 4 |
| 1002 | Мороз | Василий | Петрович | 31.12.1956 | 80295566321 | 2 | 3 |
| 1003 | Гладких | Александр | Иванович | 17.04.1982 | 80443215432 | 3 | 4 |
| 1004 | Петунин | Николай | Петрович | 12.04.1990 | 80293399432 | 3 | 4 |
| 1005 | Бамбизо | Виталий |  | 16.11.1985 | 80298765678 | 2 | 2 |
| 1006 | Федосеенко | Валерий | Васильевич | 23.12.1988 | 80299962233 | 4 | 5 |
| 1007 | Мазуров | Владимир | Васильевич | 14.0.1978 | 80296565432 | 4 | 5 |
| 1008 | Жданович | Юрий |  | 21.08.1980 | 80332457865 | 5 | 6 |
| 1009 | Бондаренко | Сергей |  | 22.03.1980 | 80331264438 | 5 | 6 |

**Естественное объединение ( INNER JOIN)**

Запрос в примере 5.6 возвращает всю информацию обо всех сотрудниках, включая названия отделов, в которых они работают, при этом для номера отдела отображаются дубликаты столбцов из разных таблиц.

Пример 5.6. Запрос для выборки данных из двух таблиц с естественным объединением

**USE Shop;**

**SELECT empId, LName, FName, Dept**

**FROM Employees E INNER JOIN Depts D**

**ON E.deptId = D.deptId;**

В примере 5.6 в инструкции SELECT для выборки указаны три столбца из таблицы Employees и один столбец из таблицы Depts. Предложение FROM инструкции SELECT определяет соединяемые таблицы, а также явно указывает тип операции объединения — INNER JOIN. Предложение ON является частью предложения FROM и указывает по значениям которых производится объединение строк в результирующем наборе.

Результат выполнения этого запроса:

|  |  |  |  |
| --- | --- | --- | --- |
| **empId** | **LName** | **FName** | **Dept** |
| 1000 | Сытый | Анатолий | Дирекция |
| 1001 | Сизов | Александр | Отдел продаж |
| 1002 | Мороз | Василий | Бухгалтерия |
| 1003 | Гладких | Александр | Отдел продаж |
| 1004 | Петунин | Николай | Отдел продаж |
| 1005 | Бамбизо | Виталий | Бухгалтерия |
| 1006 | Федосеенко | Валерий | Склад |
| 1007 | Мазуров | Владимир | Склад |
| 1008 | Жданович | Юрий | Транспортный отдел |
| 1009 | Бондаренко | Сергей | Транспортный отдел |

Соединяемые столбцы не обязательно должны иметь одинаковое имя, но должны иметь совместимые типы данных, например, INT и SMALLINT.

На рисунке 5.2 приведено графическое пояснение выбора и объединения строк таблиц при естественном объединении.

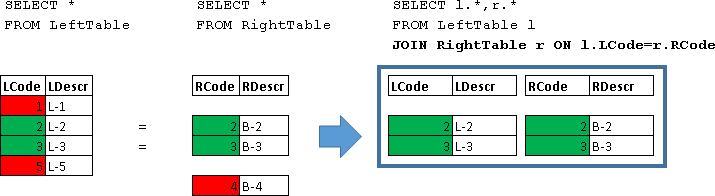


Рисунок 5.2 Выбор и объединение строк таблиц при естественном объединении.

**Объединение более чем двух таблиц**

Теоретически количество таблиц, которые можно соединить в инструкции SELECT, неограниченно. (Но одно условие объединения совмещает только две таблицы) Однако для компонента Database Engine количество соединяемых таблиц в инструкции SELECT ограничено 64 таблицами.

В примере 5.7 показано объединение трех таблиц базы данных Shop.

Пример 5.7. Выборка данных о сотрудниках из трех таблиц.

**USE Shop;**

**SELECT empId, LName, FName, Dept, Position**

**FROM employees E INNER JOIN depts D**

**ON E.deptId = D.deptId**

**INNER JOIN Positions P ON E.posId=P.posId;**

Результат выполнения этого запроса:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **empId** | **LName** | **FName** | **Dept** | **Position** |
| 1000 | Сытый | Анатолий | Дирекция | Директор |
| 1001 | Сизов | Александр | Отдел продаж | Продавец |
| 1002 | Мороз | Василий | Бухгалтерия | Бухгалтер |
| 1003 | Гладких | Александр | Отдел продаж | Продавец |
| 1004 | Петунин | Николай | Отдел продаж | Продавец |
| 1005 | Бамбизо | Виталий | Бухгалтерия | Главный бухгалтер |
| 1006 | Федосеенко | Валерий | Склад | Грузчик |
| 1007 | Мазуров | Владимир | Склад | Грузчик |
| 1008 | Жданович | Юрий | Транспортный отдел | Водитель |
| 1009 | Бондаренко | Сергей | Транспортный отдел | Водитель |

Для осуществления естественного объединения трех таблиц используется два условия объединения, каждое из которых соединяет по две таблицы.

**Внешнее объединение (OUTER JOIN)**

В предшествующих примерах естественного объединения результирующий набор содержал только те строки из одной таблицы, для которых имелись соответствующие строки в другой таблице. Но иногда кроме совпадающих строк бывает необходимым извлечь из одной или обеих таблиц строки без совпадений. Такая операция называется **внешним объединением (outer join).**

В примерах 5.8 и 5.9 показаны запросы с внешним объединением таблиц

Пример 5.8. Запрос на выборку информации о сотрудниках с внешним левым объединением таблиц.

**USE Shop;**

**SELECT empId, LName, FName, E.deptId, D.deptId, Dept**

**FROM Employees E**

**LEFT JOIN Depts D ON E.deptId=D.deptId**

Результат выполнения этого запроса:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **empId** | **LName** | **FName** | **E.deptId** | **D.DeptId** | **Dept** |
| 1000 | Сытый | Анатолий | 1 | 1 | Дирекция |
| 1001 | Сизов | Александр | 3 | 3 | Отдел продаж |
| 1002 | Мороз | Василий | 2 | 2 | Бухгалтерия |
| 1003 | Гладких | Александр | 3 | 3 | Отдел продаж |
| 1004 | Петунин | Николай | 3 | 3 | Отдел продаж |
| 1005 | Бамбизо | Виталий | 2 | 2 | Бухгалтерия |
| 1006 | Федосеенко | Валерий | 4 | 4 | Склад |
| 1007 | Мазуров | Владимир | 4 | 4 | Склад |
| 1008 | Жданович | Юрий | 5 | 5 | Транспортный отдел |
| 1009 | Бондаренко | Сергей | 5 | 5 | Транспортный отдел |

В результате левого внешнего объединения в таблице появились названия отделов всех сотрудников. Но если в таблице Employees одному из сотрудников (например, Мазурову Владимиру) в поле deptId поместить значение NULL, то результат запроса 5.9 примет следующий вид:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **empId** | **LName** | **FName** | **E.deptId** | **D.DeptId** | **Dept** |
| 1000 | Сытый | Анатолий | 1 | 1 | Дирекция |
| 1001 | Сизов | Александр | 3 | 3 | Отдел продаж |
| 1002 | Мороз | Василий | 2 | 2 | Бухгалтерия |
| 1003 | Гладких | Александр | 3 | 3 | Отдел продаж |
| 1004 | Петунин | Николай | 3 | 3 | Отдел продаж |
| 1005 | Бамбизо | Виталий | 2 | 2 | Бухгалтерия |
| 1006 | Федосеенко | Валерий | 4 | 4 | Склад |
| 1007 | Мазуров | Владимир | NULL | NULL |  |
| 1008 | Жданович | Юрий | 5 | 5 | Транспортный отдел |
| 1009 | Бондаренко | Сергей | 5 | 5 | Транспортный отдел |

Как видно из приведенного результата строка с идентификатором 1007 в результирующей таблице присутствует полностью, а присоединенная строка из правой таблицы заполнена значениями NULL.

На рисунке 5.3 приведено графическое пояснение выбора и объединения строк таблиц при внешнем левом объединении.

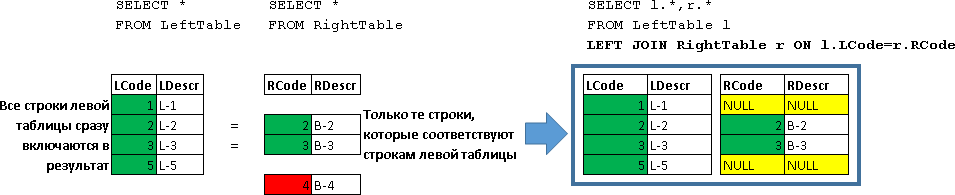


Рисунок 5.3 Выбор и объединение строк таблиц при внешнем левом объединении.

Если из таблиц Employees (таблица 5.3) удалить строки со значениями empId=1008 и empId=1009, то результат внешнего правого объединения таблиц Employees и Depts по полю deptId будет иметь вид как показано в примере 5.10.

Пример 5.10. Запрос на выборку информации о сотрудниках с внешним правым объединением таблиц.

**USE Shop;**

**SELECT empId, LName, FName, E.deptId, D.deptId, Dept**

**FROM Employees E**

**LEFT JOIN Depts D ON E.deptId=D.deptId**

Результат выполнения этого запроса:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **empId** | **LName** | **FName** | **E.deptId** | **D.DeptId** | **Dept** |
| 1000 | Сытый | Анатолий | 1 | 1 | Дирекция |
| 1001 | Сизов | Александр | 3 | 3 | Отдел продаж |
| 1002 | Мороз | Василий | 2 | 2 | Бухгалтерия |
| 1003 | Гладких | Александр | 3 | 3 | Отдел продаж |
| 1004 | Петунин | Николай | 3 | 3 | Отдел продаж |
| 1005 | Бамбизо | Виталий | 2 | 2 | Бухгалтерия |
| 1006 | Федосеенко | Валерий | 4 | 4 | Склад |
| 1007 | Мазуров | Владимир | 4 | 4 | Склад |
| NULL | NULL | NULL | NULL | 5 | Транспортный отдел |

Как видно из приведенного результата в последней строке результирующей таблицы строка из правой таблицы присутствует полностью, а присоединенная строка из левой таблицы заполнена значениями NULL.

На рисунке 5.4 приведено графическое пояснение выбора и объединения строк таблиц при внешнем правом объединении.

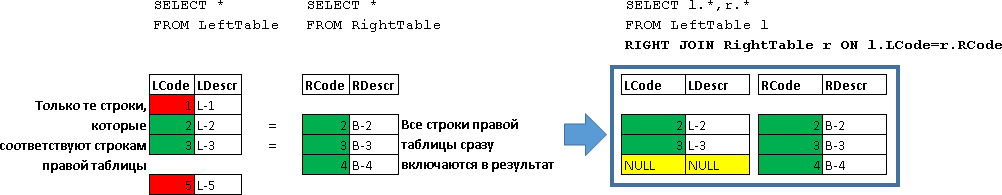


Рисунок 5.4 Выбор и объединение строк таблиц при внешнем правом объединении.

Кроме левого и правого внешнего объединения, также существует полное внешнее объединение, которое является объединением левого и правого внешних объединений. Иными словами, результирующий набор такого объединения состоит из всех строк обеих таблиц. Если для строки одной из таблиц нет соответствующей строки в другой таблице, всем ячейкам строки второй таблицы присваивается значение NULL. Для выполнения операции правого внешнего объединения используется оператор FULL OUTER JOIN.

Предположим, что в таблице Employees (таблица 5.3) удалены строки со значениями empId=1008 и empId=1009, а в таблице Depts (таблица 5.1) удалена строка со значением deptId=4 (Склад). В этом случае запрос с полным внешним объединением по полю deptId вернет результат, показанный в примере 5.11.

Пример 5.11. Запрос на выборку информации о сотрудниках с внешним полным объединением таблиц.

**USE Shop;**

**SELECT empId, LName, FName, E.deptId, D.deptId, Dept**

**FROM Employees E**

**FULL JOIN Depts D ON E.deptId=D.deptId**

Результат выполнения этого запроса:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **empId** | **LName** | **FName** | **E.deptId** | **D.DeptId** | **Dept** |
| 1000 | Сытый | Анатолий | 1 | 1 | Дирекция |
| 1001 | Сизов | Александр | 3 | 3 | Отдел продаж |
| 1002 | Мороз | Василий | 2 | 2 | Бухгалтерия |
| 1003 | Гладких | Александр | 3 | 3 | Отдел продаж |
| 1004 | Петунин | Николай | 3 | 3 | Отдел продаж |
| 1005 | Бамбизо | Виталий | 2 | 2 | Бухгалтерия |
| 1006 | Федосеенко | Валерий | 4 | NULL | NULL |
| 1007 | Мазуров | Владимир | 4 | NULL | NULL |
| NULL | NULL | NULL | NULL | 5 | Транспортный отдел |

Как видно из приведенного результата в результирующей таблице появились три не полностью заполненные строки – две строки, заполненные значениями NULL справа и одна, заполненная значениями NULL слева.

На рисунке 5.5 приведено графическое пояснение выбора и объединения строк таблиц при внешнем полном объединении.

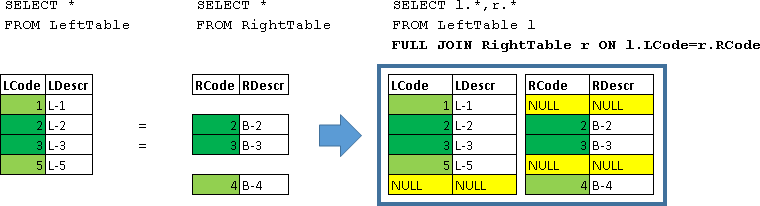


Рисунок 5.5 Выбор и объединение строк таблиц при внешнем полном объединении.

**Декартово произведение (CROSS JOIN)**

Запрос для создания объединения таблиц employee и department, используя декартово произведение, показан в примере 5.12.

Пример 5.12. Объединение таблиц декартовым произведением

**USE Shop;**

**SELECT empId, LName, FName, E.deptId, D.deptId, Dept**

**FROM Employees E**

**CROSS JOIN Depts D**

Результирующий набор выполнения примера 5.12 показан ниже.

Декартово произведение соединяет каждую строку первой таблицы с каждой строкой второй. В общем, результатом декартового произведения первой таблицы с n строками и второй таблицы с m строками будет таблица с n × m строками. Таким образом, результирующий набор запроса в примере 5.12 имеет 9 × 5 = 45 строк.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **empId** | **empLast Name** | **emp First Name** | **E.deptId** | **D.DeptId** | **dept** |
| 1000 | Сытый | Анатолий | 1 | 1 | Дирекция |
| 1001 | Сизов | Александр | 3 | 1 | Дирекция |
| 1002 | Мороз | Василий | 2 | 1 | Дирекция |
| 1003 | Гладких | Александр | 3 | 1 | Дирекция |
| 1004 | Петунин | Николай | 3 | 1 | Дирекция |
| 1005 | Бамбизо | Виталий | 2 | 1 | Дирекция |
| 1006 | Федосеенко | Валерий | 4 | 1 | Дирекция |
| 1007 | Мазуров | Владимир | 4 | 1 | Дирекция |
| 1008 | Жданович | Юрий | 5 | 1 | Дирекция |
| 1009 | Бондаренко | Сергей | 5 | 1 | Дирекция |
| 1000 | Сытый | Анатолий | 1 | 2 | Бухгалтерия |
| 1001 | Сизов | Александр | 3 | 2 | Бухгалтерия |
| 1002 | Мороз | Василий | 2 | 2 | Бухгалтерия |
| 1003 | Гладких | Александр | 3 | 2 | Бухгалтерия |
| 1004 | Петунин | Николай | 3 | 2 | Бухгалтерия |
| 1005 | Бамбизо | Виталий | 2 | 2 | Бухгалтерия |
| 1006 | Федосеенко | Валерий | 4 | 2 | Бухгалтерия |
| 1007 | Мазуров | Владимир | 4 | 2 | Бухгалтерия |
| 1008 | Жданович | Юрий | 5 | 2 | Бухгалтерия |
| 1009 | Бондаренко | Сергей | 5 | 2 | Бухгалтерия |
| 1000 | Сытый | Анатолий | 1 | 3 | Отдел продаж |
| 1001 | Сизов | Александр | 3 | 3 | Отдел продаж |
| 1002 | Мороз | Василий | 2 | 3 | Отдел продаж |
| 1003 | Гладких | Александр | 3 | 3 | Отдел продаж |
| 1004 | Петунин | Николай | 3 | 3 | Отдел продаж |
| 1005 | Бамбизо | Виталий | 2 | 3 | Отдел продаж |
| 1006 | Федосеенко | Валерий | 4 | 3 | Отдел продаж |
| 1007 | Мазуров | Владимир | 4 | 3 | Отдел продаж |
| 1008 | Жданович | Юрий | 5 | 3 | Отдел продаж |
| 1009 | Бондаренко | Сергей | 5 | 3 | Отдел продаж |
| 1000 | Сытый | Анатолий | 1 | 4 | Склад |
| 1001 | Сизов | Александр | 3 | 4 | Склад |
| 1002 | Мороз | Василий | 2 | 4 | Склад |
| 1003 | Гладких | Александр | 3 | 4 | Склад |
| 1004 | Петунин | Николай | 3 | 4 | Склад |
| 1005 | Бамбизо | Виталий | 2 | 4 | Склад |
| 1006 | Федосеенко | Валерий | 4 | 4 | Склад |
| 1007 | Мазуров | Владимир | 4 | 4 | Склад |
| 1008 | Жданович | Юрий | 5 | 4 | Склад |
| 1009 | Бондаренко | Сергей | 5 | 4 | Склад |
| 1000 | Сытый | Анатолий | 1 | 5 | Транспортный отдел |
| 1001 | Сизов | Александр | 3 | 5 | Транспортный отдел |
| 1002 | Мороз | Василий | 2 | 5 | Транспортный отдел |
| 1003 | Гладких | Александр | 3 | 5 | Транспортный отдел |
| 1004 | Петунин | Николай | 3 | 5 | Транспортный отдел |
| 1005 | Бамбизо | Виталий | 2 | 5 | Транспортный отдел |
| 1006 | Федосеенко | Валерий | 4 | 5 | Транспортный отдел |
| 1007 | Мазуров | Владимир | 4 | 5 | Транспортный отдел |
| 1008 | Жданович | Юрий | 5 | 5 | Транспортный отдел |
| 1009 | Бондаренко | Сергей | 5 | 5 | Транспортный отдел |

На практике декартово произведение применяется крайне редко.

На рисунке 5.6 приведено графическое пояснение выбора и объединения строк таблиц при декартовом произведении.

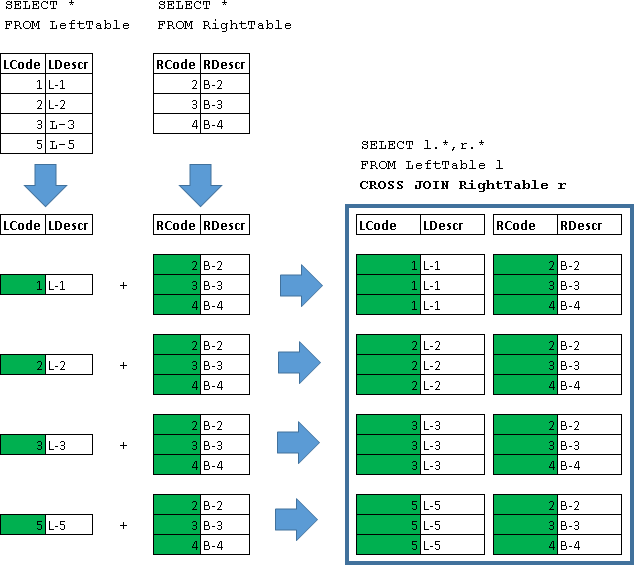


Рисунок 5.6 Выбор и объединение строк таблиц при декартовом произведении.

***Задание на лабораторную работу:***

**Вариант 1**

Используя операции соединения таблиц построить следующие запросы:

1. Составить полные сведения о читателях и книгах, которые за ними числятся. Вывести полную информацию о читателях и книгах.

2. Вывести полную информацию о книгах, которые числятся за данным читателем

3. Вывести полную информацию о читателях, которые взяли книги, изданные в прошедшем году.

4. Составить список должников на сегодняшний день.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Отыскать какие книги изданы разными издательствами.

2. Определить есть ли среди читателей и авторов однофамильцы

3. Отыскать книги, которые в текущем году еще не брали читатели.

**Вариант 2**

Используя операции соединения таблиц построить следующие запросы:

1. Составить полные сведения о клиентах и карточках, которые за ними числятся. Вывести полную информацию о клиентах и карточках.

2. Вывести полную информацию о карточках, которые числятся за данным клиентом

3. Вывести полную информацию о клиентах, которые имеют карточки, срок действия  которых заканчивается в следующем году.

4. Составить список клиентов, срок действия карточек которых истек.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список клиентов, с карточками которых не производились никакие операции.

2. Получить информацию о типах карточек и суммах на них, а также информацию о карточках и суммах операций

3. Определить номера карточек, с которыми проводились операции снятия заданной суммы.

**Вариант 3**

Используя операции соединения таблиц построить следующие запросы:

1. Для каждого подразделения вывести полную информацию о сотрудниках, их должностях и заработной плате.
2. Для заданного подразделения составить штатное расписание(должность и количество сотрудников).
3. Составить список сотрудников, работающих в заданной должности.
4. Составить список сотрудников, уволенных в прошлом году.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список подразделений, в которых совпадают названия должностей.
2. Составить список подразделений, в которых  имеются все должности.
3. Составить список сотрудников-однофамильцев.

**Вариант 4**

Используя операции соединения таблиц построить следующие запросы:

1. Составить список студентов заданной группы.
2. Составить список студентов с указанием номера группы, зачисленных приказом с заданным номером.
3. Составить список старост заданного факультета.
4. Определить название факультета по номеру группы.

 Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список студентов, фамилии которых есть на факультете ФИТР, но отсутствуют на факультете АТФ.
2. Составить список студентов третьего курса.
3. Составить список студентов-однофамильцев.

**Вариант 5**

Используя операции соединения таблиц построить следующие запросы:

1. Составить полную информацию о поездках водителя с заданной фамилией.
2. Получить полную информацию о транспорте и водителях, задействованных в поездках на прошлой неделе.
3. Составить список марок автомобилей, эксплуатирующихся более 10 лет.
4. Составить список водителей, выезжавших на автомобиле заданной марки в прошлом месяце.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Найти однофамильцев среди водителей.

2. Составить список автомобилей, выезжавших в рейс с интервалом менее двух дней.

3. Составить список автомобилей одновременно выезжавших на прошлой неделе.

**Вариант 6**

Используя операции соединения таблиц построить следующие запросы:

1. Составить список номеров второго этажа с указанием их типа.

2. Составить список клиентов, проживающих на третьем этаже с указанием номера и его типа.

3. Составить таблицу полной информации о номерах отеля.

4. Составить таблицу, содержащую сведения о номерах и клиентах на следующую неделю.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Найти однофамильцев среди клиентов.
2. Составить список клиентов, которые ранее проживали в отеле и забронировали номера на следующую неделю.
3. Составить список клиентов, несколько раз проживавших в одном и том же номере.

**Вариант 7**

Используя операции соединения построить следующие запросы:

1. Составить список врачей с указанием их специальности.
2. Составить полный график приема (с указанием врачей, специальностей, пациентов) на завтра.
3. Составить график приема врачей-терапевтов на следующую неделю.
4. Составить список врачей с указанием специальности, которых посетил пациент Иванов И.И. на прошлой неделе.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список пациентов-однофамильцев
2. Составить список пациентов посетивших разных врачей одной специальности на прошлой неделе.
3. Вывести список пациентов, у которых совпадает время посещения одного и того же врача.

**Вариант 8**

Используя операции соединения построить следующие запросы:

1. Составить список заказов с указанием входящих в них товаров на завтра.
2. Составить список адресов доставки заказов на завтра.
3. Составить список клиентов, которым были доставлены заказы вчера
4. Составить список товаров с указанием количества и цены для комплектования заказов на сегодня.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список клиентов, которые делали заказы более 1 раза за прошедший месяц.
2. Найти и объединить заказы одного и того же клиента на сегодня.
3. Составить список клиентов-однофамильцев.

**Вариант 9**

Используя операции соединения построить следующие запросы:

1. Составить список рейсов, вылетающих из Минска с указанием аэропорта прибытия.
2. Составить список клиентов, которые вчера вылетали в Амстердам.
3. Вывести список продаж за сегодня с указанием номеров рейса, цены, аэропортов отправки и прибытия, данных клиента.
4. Составить список рейсов, прибывающих в Минск сегодня.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список клиентов, которые приобрели более 1 билета за прошедший месяц.
2. Составить список клиентов, которые приобретали билеты на рейсы, вылетавшие на прошлой неделе и приобрели билеты на рейсы вылетающие на следующей неделе.
3. Составить список клиентов-однофамильцев.

**Вариант 10**

Используя операции соединения построить следующие запросы:

1. Составить список преподавателей с указанием преподаваемых предметов.

2. Составить список учеников и их оценок по математике за прошлую неделю.

3. Составить список оценок с указанием дат и предметов, полученных учеником Ивановым.

4. Составить список выставленных преподавателем Петровым оценок с указанием дат.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список учеников, которые получили более 3 оценок за прошедшую неделю.

2. Составить список учеников, получивших более 8 баллов на прошлой неделе и вчера.

3. Составить список учеников-однофамильцев.

**Вариант 11**

Используя операции соединения построить следующие запросы:

1. Составить репертуар театра на следующий месяц.

2. Определить количество проданных мест на спектакль «Отелло» на сегодня.

3. Определить сумму выручки от спектакля «Отелло» за прошлый месяц.

4. Составить список дат, когда в театре состоится спектакль «Отелло».

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список спектаклей, которые состоялись в прошлом месяце более 3 раз.

2. Определить нет ли среди проданных билетов на спектакль «Отелло» на сегодня билетов, проданных на одно и то же место.

3. Определить нет ли в афише спектаклей, назначенных на одно и то же время.

**Вариант 12**

Используя операции соединения построить следующие запросы:

1. Составить список студентов, проживающих в общежитии с указанием комнат и групп.

2. Составить список студентов, проживающих в двухместных комнатах.

3. Составить список студентов, размещенных в общежитии вчера.

4. Составить список комнат, в которых размещены студенты группы 115.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список комнат, в которых проживают студенты одной и той же группы.

2. Определить нет ли в списке проживающих одного и того же студента, числящегося в разных комнатах.

3. Составить список студентов-однофамильцев.

**Вариант 13**

Используя операции соединения построить следующие запросы:

1. Составить список дисциплин, читаемых кафедрой ПОВТ.

2. Составить список модулей с указанием входящих в них дисциплин.

3. Составить список дисциплин, читаемых в первом семестре.

4. Составить список кафедр, читающих лекции в первом семестре.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список дисциплин с одинаковым количеством учебных часов.

2. Составить список дисциплин, читаемых в нескольких семестрах.

3. Определить нет ли среди в списке дисциплин повторения названий дисциплин.

**Вариант 14**

Используя операции соединения построить следующие запросы:

1. Составить список объектов недвижимости города Минска с указанием цены и типа.

2. Составить список городов, в которых продаются коттеджи стоимостью ниже 50000.

3. Составить список районов города Минска, в которых есть объекты недвижимости стоимостью менее 30000.

4. Вывести информацию (город, район, тип) о самом дешевом объекте недвижимости.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список объектов недвижимости, у которых цена одинакова.

2. Определить города с одинаковыми названиями районов.

3. Вывести список недвижимости города Бреста и Первомайского района города Минска.

**Вариант 15**

Используя операции соединения построить следующие запросы:

1. Составить список сотрудников, работающих сегодня.

2. Составить список клиентов с указанием услуг, записанных на сегодня.

3. Составить список услуг с указанием цены, запланированных на сегодня.

4. Составить расписание работы для каждого сотрудника с указанием клиентов и оказываемых услуг.

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить список услуг, у которых цена одинакова.

2. Вывести список сотрудников, которым запланировано одновременное указание услуг.

3. Вывести список клиентов-однофамильцев.

**Содержание отчета о лабораторной работе**

1. Название и цель работы.
2. Номер группы, фамилия и инициалы студента.
3. Задание на лабораторную работу.
4. Перечень запросов и результатов их выполнения.
5. Выводы.

***Контрольные вопросы.***

1. Как реализуется натуральное соединение на языке SQL?
2. В чем отличие INNER JOIN и FULL JOIN. Привести примеры
3. В чем отличие LEFT JOIN и JOIN?
4. Как выполнить соединение трех таблиц?
5. Для каких таблиц можно использовать операции UNION, EXCEPT, INTERSECT?