**Лабораторная работа № 5. Многотабличные запросы.**

Цель работы: Получить правктические навыки формирования запросов с объединением таблиц

***Краткие теоретические сведения***

Язык Transact-SQL поддерживает еще три оператора вертикального соединения таблиц:

• UNION;

• INTERSECT;

• EXCEPT.

**Оператор UNION**

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

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.

Пример 5.1. Объединение таблиц с помощью инструкции UNION

USE sample;

SELECT domicile

FROM employee

UNION

SELECT location

FROM department;

Результат выполнения этого запроса:

|  |
| --- |
| **Domicile** |
| SanAntonio |
| Houston |
| Portland |
| Tacoma |
| Seattle |
| Dallas |

Объединять с помощью инструкции UNION можно только совместимые таблицы. Под совместимыми таблицами имеется в виду, что оба списка столбцов выборки должны содержать одинаковое число столбцов, а соответствующие столбцы должны иметь совместимые типы данных. Результат объединения можно упорядочить, только используя предложение ORDER BY в последней инструкции SELECT, как это показано в примере 5.2. Предложения GROUP BY и HAVING можно применять с отдельными инструкциями SELECT, но не в самом объединении.

Пример 5.2. Упорядочивание результатов объединения двух таблиц

USE sample;

SELECT emp\_no

FROM employee

WHERE dept\_no = 'd1'

UNION

SELECT emp\_no

FROM works\_on

WHERE enter\_date< '01.01.2007'

ORDER BY 1;

Запрос в примере 5.2 осуществляет выборку сотрудников, которые или работают в отделе d1, или начали работать над проектом до 1 января 2007 г. Этот запрос возвращает следующий результат:

|  |
| --- |
| **emp\_no** |
| 9031 |
| 10102 |
| 18316 |
| 28559 |
| 29346 |

Вместо оператора UNION можно применить оператор OR, если все инструкции SELECT, соединенные одним или несколькими операторами UNION, ссылаются на

одну и ту же таблицу. В таком случае набор инструкций SELECT заменяется

одной инструкцией SELECT с набором операторов OR.

**Операторы INTERSECT и EXCEPT**

Два других оператора для работы с наборами, INTERSECT и EXCEPT, определяют пересечение и разность соответственно. Под пересечением понимается набор строк, которые принадлежат к обеим таблицам. А разность двух таблиц определяется как все значения, которые принадлежат к первой таблице и не присутствуют во второй. В примере 5.3 показано использование оператора

INTERSECT.

Пример 5.3. Применение оператора INTERSECT

USE sample;

SELECT emp\_no

FROM employee

WHERE dept\_no = 'd1'

INTERSECT

SELECT emp\_no

FROM works\_on

WHERE enter\_date< '01.01.2008';

Результат выполнения этого запроса:

|  |
| --- |
| **emp\_no** |
| 18316 |
| 28559 |

Использование оператора EXCEPT показано в примере 5.4.

Пример 5.4. Применение оператора EXCEPT

USE sample;

SELECT emp\_no

FROM employee

WHERE dept\_no = 'd3'

EXCEPT

SELECT emp\_no

FROM works\_on

WHERE enter\_date> '01.01.2008';

Результат выполнения этого запроса:

|  |
| --- |
| **emp\_no** |
| 10102 |
| 25348 |

Эти три оператора над множествами имеют разный приоритет выполнения: оператор INTERSECT имеет наивысший приоритет, за ним следует оператор EXCEPT, а оператор UNION имеет самый низкий приоритет.

**Горизонтальное объединение таблиц**

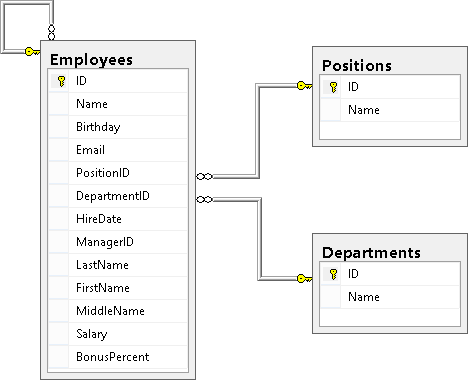
Оператор соединения таблиц JOIN позволяет извлекать данные более чем из одной таблицы. Этот оператор является наиболее важным оператором для реляционных систем баз данных, поскольку благодаря ему имеется возможность распределять данные по нескольким таблицам, обеспечивая, таким образом, важное свойство систем баз данных — отсутствие избыточности данных.

Существует пять типов соединения таблиц

1. **JOIN** – левая\_таблица JOIN правая\_таблица ON условия\_соединения.
2. **LEFT JOIN** – левая\_таблица LEFT JOIN правая\_таблица ON условия\_соединения
3. **RIGHT JOIN** – левая\_таблица RIGHT JOIN правая\_таблица ON условия\_соединения
4. **FULL JOIN** – левая\_таблица FULL JOIN правая\_таблица ON условия\_соединения
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\_соединения. |
| 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\_соединения. 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\_соединения. 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\_соединения, то они объединяются в одну строку. Для строк, для которых не выполняются условия\_соединения, 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 соединения иногда называют декартовым произведением. |

Как видно из таблицы полный синтаксис от краткого отличается только наличием слов INNER или OUTER.

В качестве примера рассмотрим БД, состоящую из 3-хтаблиц Employees, Departments и Positions. Схемабазы данных имеетвид:  


Содержимоетаблицследующее

**Таблица Employees**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **empID** | **Name** | **Birthday** | **PositionID** | **DepartmentID** | **HireDate** | **City** |
| 1000 | ИвановИ.И. | 1955-02-19 | 2 | 1 | 2015-04-08 | Гомель |
| 1001 | ПетровП.П. | 1983-12-03 | 3 | 3 | 2015-04-08 | Борисов |
| 1002 | СидоровС.С. | 1976-06-07 | 1 | 2 | 2015-04-09 | Орша |
| 1003 | АндреевА.А. | 1982-04-17 | 4 | 3 | 2015-04-12 | Минск |
| 1004 | НиколаевН.Н. | NULL | 3 | 3 | NULL | Брест |
| 1005 | АлексеевА.А. | NULL | NULL | NULL | NULL | Гродно |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Таблица Departments**   |  |  |  | | --- | --- | --- | | **DepartmentID** | **Name** | **Location** | | 1 | Администрация | Минск | | 2 | Бухгалтерия | Витебск | | 3 | ИТ | Пинск | | 4 | Маркетингиреклама | Минск | | 5 | Логистика | Минск | | **Таблица Positions**   |  |  | | --- | --- | | **positionID** | **name** | | 1 | Бухгалтер | | 2 | Директор | | 3 | Программист | | 4 | Старшийпрограммист | |

Если суть РДБ – разделяй и властвуй, то суть операций объединений снова склеить разбитые по таблицам данные, т.е. привести их обратно в человеческий вид.  
Если говорить просто, то операции горизонтального соединения таблицы с другими таблицами используются для того, чтобы получить из них недостающие данные.

**JOIN (Внутреннее соединение)**

Из строк левой таблицы и правой таблицы объединяются и возвращаются только те строки, по которым выполняются условия соединения.

-- JOIN вернет 5 строк

SELECTemp.empID, emp.Name, emp.DepartmentID, dep.DEpartmentID, dep.Name

FROMEmployees emp

JOINDepartments depON emp.DepartmentID=dep.ID

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **empID** | **Name** | **DepartmentID** | **DepartmentID** | **Name** |
| 1000 | ИвановИ.И. | 1 | 1 | Администрация |
| 1001 | ПетровП.П. | 3 | 3 | ИТ |
| 1002 | СидоровС.С. | 2 | 2 | Бухгалтерия |
| 1003 | АндреевА.А. | 3 | 3 | ИТ |
| 1004 | НиколаевН.Н. | 3 | 3 | ИТ |

**LEFTJOIN (Левое внешнее соединение)**

Возвращаются все строки левой таблицы (ключевое слово LEFT). Данными правой таблицы дополняются только те строки левой таблицы, для которых выполняются условия соединения. Для недостающих данных вместо строк правой таблицы вставляются NULL-значения.

*-- LEFT JOIN вернет 6 строк*

SELECT emp.empID,emp.Name,emp.DepartmentID,dep.ID,dep.Name

FROM Employees emp

LEFT JOIN Departments dep ON emp.DepartmentID=dep.ID

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **empID** | **Name** | **DepartmentID** | **DepartmentID** | **Name** |
| 1000 | ИвановИ.И. | 1 | 1 | Администрация |
| 1001 | ПетровП.П. | 3 | 3 | ИТ |
| 1002 | СидоровС.С. | 2 | 2 | Бухгалтерия |
| 1003 | АндреевА.А. | 3 | 3 | ИТ |
| 1004 | НиколаевН.Н. | 3 | 3 | ИТ |
| 1005 | АлексеевА.А. | NULL | NULL | NULL |

**RIGHTJOIN (Правое внешнее соединение)**

Возвращаются все строки правой таблицы (ключевое слово RIGHT). Данными левой таблицы дополняются только те строки правой таблицы, для которых выполняются условия соединения. Для недостающих данных вместо строк левой таблицы вставляются NULL-значения.

*-- RIGHT JOIN вернет 7 строк*

SELECT emp.empID,emp.Name,emp.DepartmentID,dep.ID,dep.Name

FROM Employees emp

RIGHT JOIN Departments dep ON emp.DepartmentID=dep.ID

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **empID** | **Name** | **DepartmentID** | **DepartmentID** | **Name** |
| 1000 | ИвановИ.И. | 1 | 1 | Администрация |
| 1002 | СидоровС.С. | 2 | 2 | Бухгалтерия |
| 1001 | ПетровП.П. | 3 | 3 | ИТ |
| 1003 | АндреевА.А. | 3 | 3 | ИТ |
| 1004 | НиколаевН.Н. | 3 | 3 | ИТ |
| NULL | NULL | NULL | 4 | Маркетингиреклама |
| NULL | NULL | NULL | 5 | Логистика |

**FULL JOIN (Полноесоединение)**

Возвращаются все строки левой таблицы и правой таблицы. Если для строк левой таблицы и правой таблицы выполняются условия соединения, то они объединяются в одну строку. Для строк, для которых не выполняются условия соединения, NULL-значения вставляются на место левой таблицы, либо на место правой таблицы, в зависимости от того данных какой таблицы в строке не имеется.

*-- FULL JOIN вернет 8 строк*

SELECT emp.empID,emp.Name,emp.DepartmentID,dep.ID,dep.Name

FROM Employees emp

FULL JOIN Departments dep ON emp.DepartmentID=dep.ID

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **empID** | **Name** | **DepartmentID** | **DepartmentID** | **Name** |
| 1000 | ИвановИ.И. | 1 | 1 | Администрация |
| 1001 | ПетровП.П. | 3 | 3 | ИТ |
| 1002 | СидоровС.С. | 2 | 2 | Бухгалтерия |
| 1003 | АндреевА.А. | 3 | 3 | ИТ |
| 1004 | НиколаевН.Н. | 3 | 3 | ИТ |
| 1005 | АлексеевА.А. | NULL | NULL | NULL |
| NULL | NULL | NULL | 4 | Маркетингиреклама |
| NULL | NULL | NULL | 5 | Логистика |

**CROSS JOIN (Декартовопроизведение)**

Объединение каждой строки левой таблицы со всеми строками правой таблицы.

*-- CROSS JOIN вернет 30 строк - (6 строк таблицы Employees) \* (5 строк таблицы Departments)*

SELECT emp.empID, emp.Name, emp.DepartmentID, dep.DepartmentID, dep.Name

FROM Employees emp

CROSS JOIN Departmentsdep

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **empID** | **Name** | **DepartmentID** | **DepartmentID** | **Name** |
| 1000 | Иванов И.И. | 1 | 1 | Администрация |
| 1001 | Петров П.П. | 3 | 1 | Администрация |
| 1002 | Сидоров С.С. | 2 | 1 | Администрация |
| 1003 | Андреев А.А. | 3 | 1 | Администрация |
| 1004 | Николаев Н.Н. | 3 | 1 | Администрация |
| 1005 | Алексеев А.А. | NULL | 1 | Администрация |
| 1000 | Иванов И.И. | 1 | 2 | Бухгалтерия |
| 1001 | Петров П.П. | 3 | 2 | Бухгалтерия |
| 1002 | Сидоров С.С. | 2 | 2 | Бухгалтерия |
| 1003 | Андреев А.А. | 3 | 2 | Бухгалтерия |
| 1004 | Николаев Н.Н. | 3 | 2 | Бухгалтерия |
| 1005 | Алексеев А.А. | NULL | 2 | Бухгалтерия |
| 1000 | Иванов И.И. | 1 | 3 | ИТ |
| 1001 | Петров П.П. | 3 | 3 | ИТ |
| 1002 | Сидоров С.С. | 2 | 3 | ИТ |
| 1003 | Андреев А.А. | 3 | 3 | ИТ |
| 1004 | Николаев Н.Н. | 3 | 3 | ИТ |
| 1005 | Алексеев А.А. | NULL | 3 | ИТ |
| 1000 | Иванов И.И. | 1 | 4 | Маркетинг и реклама |
| 1001 | Петров П.П. | 3 | 4 | Маркетинг и реклама |
| 1002 | Сидоров С.С. | 2 | 4 | Маркетинг и реклама |
| 1003 | Андреев А.А. | 3 | 4 | Маркетинг и реклама |
| 1004 | Николаев Н.Н. | 3 | 4 | Маркетинг и реклама |
| 1005 | Алексеев А.А. | NULL | 4 | Маркетинг и реклама |
| 1000 | Иванов И.И. | 1 | 5 | Логистика |
| 1001 | Петров П.П. | 3 | 5 | Логистика |
| 1002 | Сидоров С.С. | 2 | 5 | Логистика |
| 1003 | Андреев А.А. | 3 | 5 | Логистика |
| 1004 | Николаев Н.Н. | 3 | 5 | Логистика |
| 1005 | Алексеев А.А. | NULL | 5 | Логистика |

**Тета-соединение**

Условие сравнения столбцов соединения не обязательно должно быть равенством, но может быть любым другим сравнением. Соединение, в котором используется общее условие сравнения столбцов соединения, называется тета-соединением. В примере ниже показана операция тета-соединения, в которой используется условие "меньше чем". Данный запрос возвращает все комбинации информации о сотрудниках и отделах для тех случаев, когда место проживания сотрудника по алфавиту идет перед месторасположением любого отдела, в котором работает этот служащий:

SELECTemp.empID, emp.Name, emp.City, dep.Location

FROM Employees emp

JOIN Departments depONCity < Location

WHERELocation=’Минск’

Результат выполнения этого запроса:

|  |  |  |  |
| --- | --- | --- | --- |
| **empID** | **Name** | **City** | **Location** |
| 1000 | Иванов И.И. | Гомель | Минск |
| 1001 | Петров П.П. | Борисов | Минск |
| 1004 | Николаев Н.Н. | Брест | Минск |
| 1005 | Алексеев А.А. | Гродно | Минск |

В этом примере сравниваются соответствующие значения столбцов City и Location. В каждой строке результата значение столбца City сравнивается в алфавитном порядке с соответствующим значением столбца Location.

**Самосоединение, или соединение таблицы самой с собой**

Кроме соединения двух или больше разных таблиц, операцию естественного соединения можно применить к одной таблице. В данной операции таблица соединяется сама с собой, при этом один столбец таблицы сравнивается сам с собой. Сравнивание столбца с самим собой означает, что в предложении FROM инструкции SELECT имя таблицы употребляется дважды. Поэтому необходимо иметь возможность ссылаться на имя одной и той же таблицы дважды. Это можно осуществить, используя, по крайней мере, один псевдоним. То же самое относится и к именам столбцов в условии соединения в инструкции SELECT. Для того чтобы различить столбцы с одинаковыми именами, необходимо использовать уточненные имена.

Пример: определить фамилии сотрудников, принятых на работу в один день, но работающих в разных отделах.

SELECT emp.empID, emp.Name, emp.DepartmentID, emp.HireDate

FROM Employees emp

JOIN Employees emp2 ON emp.HireDate=emp2.HireDate

WHERE emp.DepartmentID<>emp2.DepartmentID

Результатвыполненияэтогозапроса:

|  |  |  |  |
| --- | --- | --- | --- |
| **empID** | **emp\_Name** | **DepartmentID** | **HireDate** |
| 1000 | Иванов И.И. | 1 | 20015-04-08 |
| 1001 | Петров П.П. | 3 | 20015-04-08 |

**Полусоединение**

Полусоединение похоже на естественное соединение, но возвращает только набор всех строк из одной таблицы, для которой в другой таблице есть одно или несколько совпадений. Использование полусоединения показано в примере ниже:

SELECTemp.empID, emp.Name, emp.DepartmentID

FROM Employees emp

JOIN Departments depONemp.DepartmentID=dep.DepartmentID

Результат выполнения запроса:

|  |  |  |
| --- | --- | --- |
| **empID** | **Name** | **DepartmentID** |
| 1000 | Иванов И.И. | 1 |
| 1001 | Петров П.П. | 3 |
| 1002 | Сидоров С.С. | 2 |
| 1003 | Андреев А.А. | 3 |
| 1004 | Николаев Н.Н. | 3 |

Как можно видеть, список выбора SELECT в полусоединении содержит только столбцы из таблицы Employees. Это и есть характерной особенностью операции полусоединения.

***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 списоккомнат, 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 списокдисциплин с одинаковым количеством учебных часов.

2. Составитьсписок 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 стоимостьюменее 30000.

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

Используя операции UNION, EXCEPT, INTERSECT построить следующие запросы:

1. Составить списокобъектов 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 списокуслуг, 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 LEFTJOINи JOIN?
4. Как выполнить соединение трех таблиц?
5. Для каких таблиц можно использовать операции UNION, EXCEPT, INTERSECT?