Профессиональная программа “Анализ данных на языке SQL”

Блок 5. Операции с несколькими таблицами

Содержание блока

- Объединение, пересечение и разность множеств.
- Вложенные запросы (подзапросы). Использование вложенных запросов в конструкции SELECT.
- Соединения таблиц.

В конце блока предусмотрено решение ряда практических задач.

1-ый учебный вопрос

Объединение, пересечение и разность множеств

Общие сведения: SQL оператор для работы с наборами UNION

Сцепляет результаты двух запросов в один результирующий набор.
UNION ALL - повторяющиеся строки включаются.
UNION - повторяющиеся строки исключаются.

В результате операции **UNION** сцепляются результирующие наборы двух запросов. При этом операция **UNION** не создает отдельные строки для столбцов, полученных из двух таблиц.

Ниже приведены основные правила объединения результирующих наборов двух запросов с помощью операции **UNION**:
- количество и порядок столбцов должны быть одинаковыми во всех запросах;
- типы данных должны быть совместимыми.

Простое объединение двух таблиц.

```
SELECT ProductModelID, Name
INTO dbo.Gloves
FROM Production.ProductModel
WHERE ProductModelID IN (3, 4);

SELECT ProductModelID, Name
FROM Production.ProductModel
WHERE ProductModelID NOT IN (3, 4)
UNION
SELECT ProductModelID, Name
FROM dbo.Gloves
ORDER BY Name;
```

100 %

Результаты | Сообщения

	ProductModelID	Name
1	122	All-Purpose Bike Stand
2	119	Bike Wash
3	115	Cable Lock
4	98	Chain
5	1	Classic Vest
6	2	Cycling Cap
7	121	Fender Set - Mountain
8	102	Front Brakes
9	103	Front Derailleur
10	3	Full-Finger Gloves
11	4	Half-Finger Gloves

На рисунке показан результат объединения таблиц Gloves и ProductModel. Таблица GLOVES была создана на результате выборки данных из ProductModel двух строк с ProductModelID 3 и 4. Для объединения берется выборка из таблицы ProductModel без строк с ProductModelID 3 и 4 и таблица Gloves.

Так как таблица Gloves есть копия выборки из таблицы ProductModel, то условия о полном совпадении таблиц при UNION соблюдено.

Объединение двух таблиц с использованием SELECT INTO.

```
SELECT ProductModelID, Name
INTO dbo.ProductResults
FROM Production.ProductModel
WHERE ProductModelID NOT IN (3, 4)
UNION
SELECT ProductModelID, Name
FROM dbo.Gloves;
GO

SELECT ProductModelID, Name
FROM dbo.ProductResults;
```

Результаты | Сообщения

ProductModelID	Name
1	Classic Vest
2	Cycling Cap
3	Full-Finger Gloves
4	Half-Finger Gloves
5	HL Mountain Frame
6	HL Road Frame

На рисунке показан результат объединения таблиц Gloves и ProductModel. С помощью конструкции SELECT INTO, результат объединения помещается в новую таблицу с названием ProductResults.

Таблицу ProductResults можно назвать простейшей витриной данных.

Объединение двух выборок SELECT с сортировкой.

```
SELECT ProductModelID, Name
FROM Production.ProductModel
WHERE ProductModelID NOT IN (3, 4)
UNION
SELECT ProductModelID, Name
FROM dbo.Gloves
ORDER BY Name;
```

Результаты | Сообщения

ProductModelID	Name
122	All-Purpose Bike Stand
119	Bike Wash
115	Cable Lock
98	Chain
1	Classic Vest

На рисунке показан результат объединения двух выборок – из таблицы Gloves выбираются все строки и столбцы и из таблицы ProductModel выбираются все строки, кроме ProductModelID 3 и 4.

Сортировка ORDER BY всегда будет находится в самом конце объединения.

Объединение двух выборок с условием UNION ALL.

```
SELECT ProductModelID, Name
FROM Production.ProductModel
UNION ALL
SELECT ProductModelID, Name
FROM dbo.Gloves
ORDER BY ProductModelID ASC;
```

На рисунке показан результат объединения двух выборок – из таблицы Gloves выбираются все строки, также как и из таблицы ProductModel.

Результаты | Сообщения

ProductModelID	Name
1	Classic Vest
2	Cycling Cap
3	Full-Finger Gloves
3	Full-Finger Gloves
4	Half-Finger Gloves
4	Half-Finger Gloves

Из-за применения конструкции UNION ALL, информация о перчатках (gloves) из двух таблиц дублируется. В случае применения UNION, дублирующие значения исключаются.

Общие сведения: Операторы INTERSECT и EXCEPT

Операторы возвращают различные строки, сравнивая результаты двух запросов.

Оператор EXCEPT возвращает уникальные строки из левого входного запроса, которые не выводятся правым входным запросом.
Оператор INTERSECT возвращает уникальные строки, выводимые левым и правым входными запросами.

Ниже приведены основные правила объединения результирующих наборов двух запросов с помощью операций **EXCEPT и INTERSECT** :
- количество и порядок столбцов должны быть одинаковыми во всех запросах;
- типы данных должны быть совместимыми.

Простое пересечение двух таблиц, INTERSECT.

```
SELECT ProductModelID, Name
FROM Production.ProductModel
INTERSECT
SELECT ProductModelID, Name
FROM dbo.Gloves
ORDER BY Name;
```

100 %

Результаты | Сообщения

	ProductModelID	Name
1	3	Full-Finger Gloves
2	4	Half-Finger Gloves

На рисунке показан результат объединения таблиц Gloves и ProductModel. В результатном наборе остались строки, которые есть одновременно и в таблице Gloves и в таблице ProductModel.

От перестановки выборок местами результат операции не поменяется.

Простая разность двух таблиц, EXCEPT.

```
SELECT ProductModelID, Name
FROM Production.ProductModel
EXCEPT
SELECT ProductModelID, Name
FROM dbo.Gloves
ORDER BY ProductModelID ASC;
```

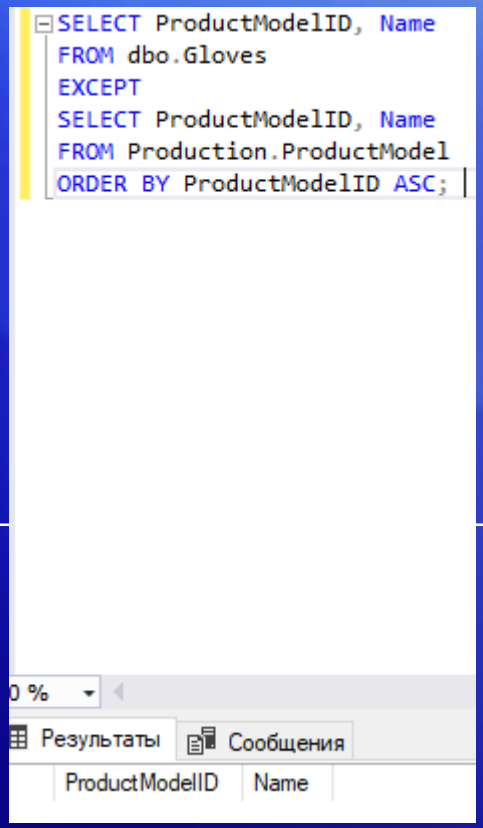
Результаты | Сообщения

ProductModelID	Name
1	Classic Vest
2	Cycling Cap
5	HL Mountain Frame
6	HL Road Frame

На рисунке показан результат исключения из таблицы ProductModel строк, которые есть и в этой таблице и в таблице Gloves.

Записи 3 и 4 исключены из результатного набора.

Простая разность двух таблиц, EXCEPT.



На картинке показан результат исключения из таблицы Gloves строк, которые есть и в этой таблице и в таблице ProductModel. Так как в таблице Gloves нет значений, отличных от значений в таблице ProductModel, результатом исключения является пустая таблица.

Как видно из примера, при выполнении исключения EXCEPT позиция таблиц относительно друг друга имеет ключевое значение.

2-ый учебный вопрос

Вложенные запросы (подзапросы). Использование вложенных запросов в конструкции SELECT

Общие сведения: Вложенные запросы

Вложенными запросами называются комбинированные операторы выбора, вставки/удаления/добавления данных, которые содержат в себе несколько подзапросов (внешние и внутренние – outer и inner).

Являются довольно изящным и простым решением при оптимизации работы БД. Чаще всего используются при вставке (INSERT), удалении (DELETE) или же изменении (UPDATE) большого количества значений в одной таблице на основании данных другой таблицы.

Общие сведения: Правила вложенных запросов

Список выбора вложенного запроса, начинающийся с оператора сравнения, может включать только одно выражение или имя столбца.

Типы данных ntext, text и image не могут быть использованы в списке выбора вложенных запросов.

Вложенные запросы, представленные оператором неизмененного сравнения (после которого нет ключевого слова ANY или ALL), не могут включать предложения GROUP BY и HAVING.
Ключевое слово DISTINCT не может быть использовано во вложенном запросе, включающем предложение GROUP BY.

Предложение ORDER BY может быть указано только вместе с предложением TOP.

Множественный уровень вложенности Тип 1.

```
SELECT LastName, FirstName
FROM Person.Person
WHERE BusinessEntityID IN
    (SELECT BusinessEntityID
     FROM HumanResources.Employee
     WHERE BusinessEntityID IN
        (SELECT BusinessEntityID
         FROM Sales.SalesPerson));
```

100 %

Результаты | Сообщения

	LastName	FirstName
1	Jiang	Stephen
2	Blythe	Michael
3	Mitchell	Linda
4	Carson	Jillian
5	Vargas	Garrett
6	Reiter	Tsvi

На картинке показан результат последовательного выполнения трех вложенных запросов: выбрать фамилию и имя всех продавцов, которые есть в таблице Employee.

Вложенные запросы будут раскрываться от последнего к первому. Количество вложений при этом не ограничено.

Множественный уровень вложенности Тип 1. Самосоединение с псевдонимами.

```
SELECT e1.StateProvinceID, e1.AddressID
FROM Person.Address AS e1
WHERE e1.AddressID IN
    (SELECT e2.AddressID
     FROM Person.Address AS e2
     WHERE e2.StateProvinceID = 39);
```

Результаты | Сообщения

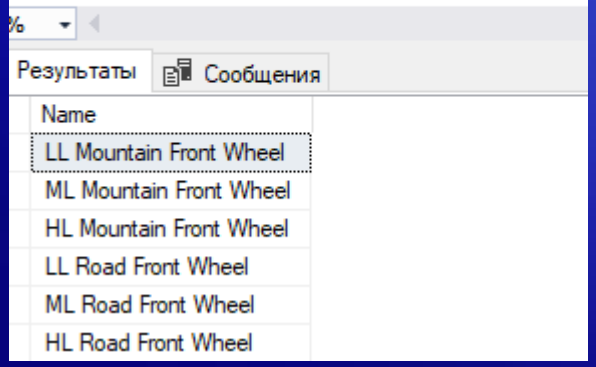
StateProvinceID	AddressID
39	942
39	955
39	972
39	22660

На картинке показан результат последовательного выполнения двух вложенных запросов к одной и той же таблице – сперва осуществляется фильтрация строк по StateProvinceID, а затем, по результату запроса осуществляется фильтрация по столбцам StateProvinceID и Address.

Переменные e1 и e2 называются псевдонимами и присваиваются программистом SQL. Они нужны для идентификации одинаковых элементов в разных вложенных запросах.

Вложенные запросы Тип 1 с ключевым словом IN.

```
SELECT Name
FROM Production.Product
WHERE ProductSubcategoryID IN
    (SELECT ProductSubcategoryID
     FROM Production.ProductSubcategory
     WHERE Name = 'Wheels');
```



На картинке показан результат последовательного выполнения двух вложенных запросов: выборка подкатегории продуктов “Wheels” (колеса) и последующая выборка на полученных значениях названия этих колес из таблицы Product.

Противоположное ключевому слову IN слово NOT IN выведет названия всех неКолес.

Вложенные запросы Тип 1 с фильтрацией WHERE.

```
SELECT Name
FROM Purchasing.Vendor
WHERE CreditRating = 1
AND BusinessEntityID IN
    (SELECT BusinessEntityID
     FROM Purchasing.ProductVendor
     WHERE MinOrderQty >= 20
     AND AverageLeadTime < 16);
```

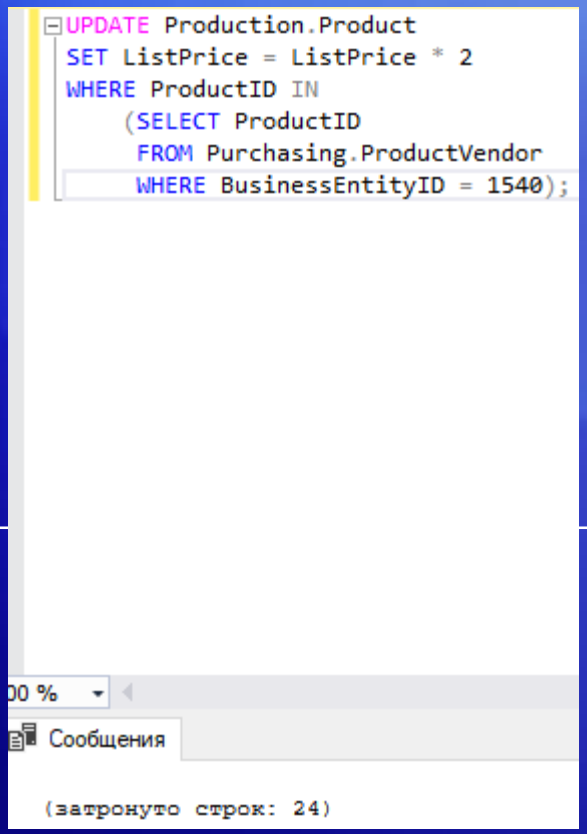
Результаты | Сообщения

Name
Compete Enterprises, Inc
International Trek Center
First National Sport Co.
Comfort Road Bicycles
Circuit Cycles
First Rate Bicycles

На картинке показан результат последовательного выполнения двух вложенных запросов: выборка поставщиков у которых было заказано не менее 20 позиций товара со сроком поставки менее 16 дней и проекция этой выборки на таблицу Vendor (поставщик) с условием кредитный рейтинг = 1.

Ограничений к WHERE во вложенных запросах нет.

Вложенные запросы Тип 2.



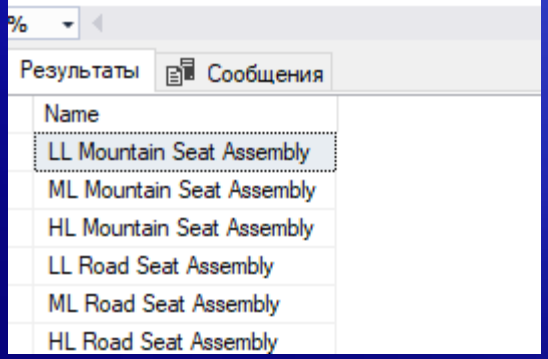
На картинке показан результат последовательного выполнения двух вложенных запросов: на основании выборки из таблицы ProductID по поставщику с номером 1540, цена в таблице продукт (Product) на эти позиции удвоится.

Вложенные запросы Тип 2 это комбинация инструкции на INSERT/UPDATE/DELETE и инструкции на выборку SELECT.

Операции сравнения с модификатором ANY

```
SELECT Name
FROM Production.Product
WHERE ListPrice >= ANY
    (SELECT MAX (ListPrice)
     FROM Production.Product
     GROUP BY ProductSubcategoryID);
```

На картинке показан запрос, возвращающий все названия продуктов из таблицы Product, цена на которые больше или равна максимальной цене по подкатегориям продуктов из таблицы Product.



Продукты также сгруппированы по подкатегориям с помощью GROUP BY.

Вложенные запросы с ключевым словом EXISTS

```
SELECT Name
FROM Production.Product
WHERE EXISTS
    (SELECT *
     FROM Production.ProductSubcategory
     WHERE ProductSubcategoryID =
           Production.Product.ProductSubcategoryID
       AND Name = 'Wheels');
```

Результаты | Сообщения

Name
LL Mountain Front Wheel
ML Mountain Front Wheel
HL Mountain Front Wheel
LL Road Front Wheel
ML Road Front Wheel
HL Road Front Wheel

На картинке показан запрос, возвращающий все названия продуктов из таблицы Product из подкатегории (столбец Subcategory) с условием Название = Колеса (Wheels).

Существует обратный EXISTS оператор NOT EXISTS.

Вложенные запросы с выражениями

```
SELECT Name, ListPrice,
(SELECT AVG(ListPrice) FROM Production.Product) AS Average,
    ListPrice - (SELECT AVG(ListPrice) FROM Production.Product)
    AS Difference
FROM Production.Product
WHERE ProductSubcategoryID = 1;
```

Результаты | Сообщения

Name	ListPrice	Average	Difference
Mountain-100 Silver, 38	6799,98	878,0071	5921,9729
Mountain-100 Silver, 42	6799,98	878,0071	5921,9729
Mountain-100 Silver, 44	6799,98	878,0071	5921,9729
Mountain-100 Silver, 48	6799,98	878,0071	5921,9729
Mountain-100 Black, 38	6749,98	878,0071	5871,9729

На картинке показан запрос, находящий цену на продукты из субкатегории 1 (велосипеды), и выводящий также среднюю цену по данной субкатегории и разницу между средней ценой и ценой этой модели велосипеда.

Агрегатные результаты выводятся во временных столбцах.

3-ий учебный вопрос

Соединения таблиц

Общие сведения: Соединения таблиц JOIN

Операция соединения, предназначена для обеспечения выборки данных из двух таблиц и включения этих данных в один результирующий набор. Отличительными особенностями операции соединения являются следующее:
- в схему таблицы-результата входят столбцы обеих исходных таблиц (таблиц-операндов), то есть схема результата является «сцеплением» схем операндов;
- каждая строка таблицы-результата является «сцеплением» строки из одной таблицы-операнда со строкой второй таблицы-операнда.

Определение того, какие именно исходные строки войдут в результат и в каких сочетаниях, зависит от типа операции соединения и от явно заданного ***условия соединения***.
При необходимости соединения не двух, а нескольких таблиц, операция соединения применяется несколько раз (последовательно).

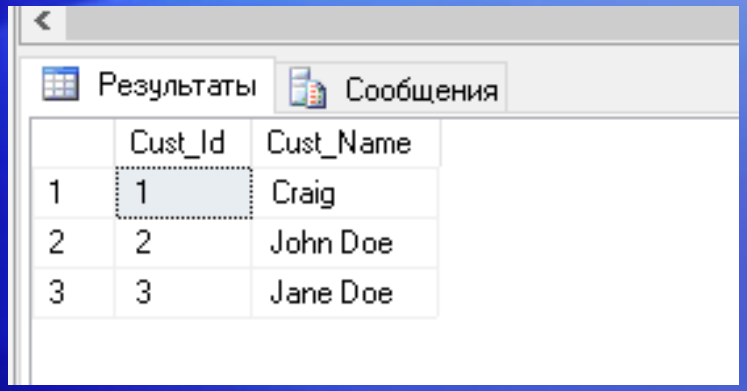
Общие сведения: Типы соединений

Небольшое предупреждение и корректировка. Приведенные типы соединений субъективны. Связано это с тем, что данные соединения используются в СУБД MS SQL Server и могут несколько отличаться синтаксисом от соединений в других СУБД.

Внутреннее соединение - Inner join (IJ)
Внешнее соединение - Outer join (OJ)
Перекрестное соединение - Cross join (CJ)
Полусоединение - Semi-join (SJ)
Анти-полусоединение - Anti-semi-join (ASJ)

Соединения таблиц JOIN

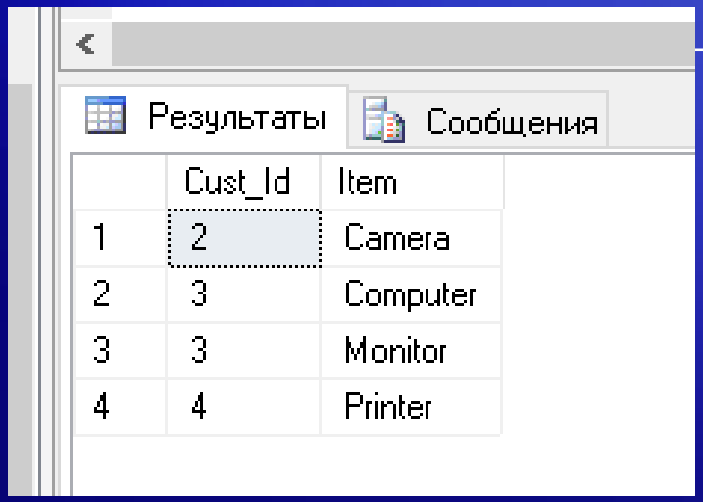
Customers (покупатели)



	Cust_Id	Cust_Name
1	1	Craig
2	2	John Doe
3	3	Jane Doe

На рисунке показаны две таблицы, которые далее будут использованы для демонстрации соединений.

Sales (продажи)



	Cust_Id	Item
1	2	Camera
2	3	Computer
3	3	Monitor
4	4	Printer

Условием эквивалентности для соединения таблиц является наличие одинакового столбца Cust_ID в обеих таблицах.

Внутреннее соединение, INNER JOIN

```
select *
from Sales S inner join Customers C
on S.Cust_Id = C.Cust_Id
```

Результаты | Сообщения

	Cust_Id	Item	Cust_Id	Cust_Name
1	2	Camera	2	John Doe
2	3	Computer	3	Jane Doe
3	3	Monitor	3	Jane Doe

Внутреннее соединение находит пары строк, которые соединяются и удовлетворяют предикату соединения. Так, показанный на картинке запрос использует предикат соединения "S.Cust_Id = C.Cust_Id", позволяющий найти все продажи и сведения о клиенте с одинаковыми значениями Cust_Id.

Таблицы кодируются псевдонимами (S-Sales, C-Customers) с целью упрощения процесса идентификации предиката

Внешнее соединение, OUTER JOIN

```
select *
from Sales S left outer join Customers C
on S.Cust_Id = C.Cust_Id
```

Результаты | Сообщения

	Cust_Id	Item	Cust_Id	Cust_Name
1	2	Camera	2	John Doe
2	3	Computer	3	Jane Doe
3	3	Monitor	3	Jane Doe
4	4	Printer	NULL	NULL

Соединение двух таблиц, в результат которого в обязательном порядке входят строки либо одной, либо обеих таблиц. Бывает левое внешнее соединение, правое внешнее соединение или же полное внешнее соединение. На картинке показан случай левого внешнего соединения.

Сервер возвращает вместо данных о клиенте значение NULL, поскольку для проданного товара 'Printer' нет соответствующей записи клиента.

Полное внешнее соединение, FULL OUTER JOIN

```
select *
from Sales S full outer join Customers C
on S.Cust_Id = C.Cust_Id
```

Результаты | Сообщения

	Cust_Id	Item	Cust_Id	Cust_Name
1	2	Camera	2	John Doe
2	3	Computer	3	Jane Doe
3	3	Monitor	3	Jane Doe
4	4	Printer	NULL	NULL
5	NULL	NULL	1	Craig

Полное внешнее соединение, выводит результат сцепления таблиц вне зависимости от их положения друг относительно друга, в обоих направлениях. Так, на рисунке показан запрос, выводящий информацию о всех клиентов (независимо от того, покупали ли они что-нибудь), и о всех продажах (независимо от того, сопоставлен ли им имеющийся клиент)

По сути, к результату LEFT OUTER JOIN добавляется результат RIGHT OUTER JOIN без дубликатов строк.

Перекрестное соединение, CROSS JOIN

```
select *
from Sales S cross join Customers C
```

	Cust_Id	Item	Cust_Id	Cust_Name
1	2	Camera	1	Craig
2	3	Computer	1	Craig
3	3	Monitor	1	Craig
4	4	Printer	1	Craig
5	2	Camera	2	John Doe
6	3	Computer	2	John Doe
7	3	Monitor	2	John Doe
8	4	Printer	2	John Doe
9	2	Camera	3	Jane Doe
10	3	Computer	3	Jane Doe
11	3	Monitor	3	Jane Doe
12	4	Printer	3	Jane Doe

Перекрестное соединение выполняет полное Декартово произведение двух таблиц. То есть это соответствие каждой строки одной таблицы - каждой строке другой таблицы. Для перекрестного соединения ***нельзя*** определить предикат соединения, используя для этого предложение ON.

Перекрестные соединения используются довольно редко. Не стоит без сильной на то надобности пересекать две очень большие таблицы, поскольку это неоправданно дорогие по затратам вычислительных мощностей операции.