Базы Данных

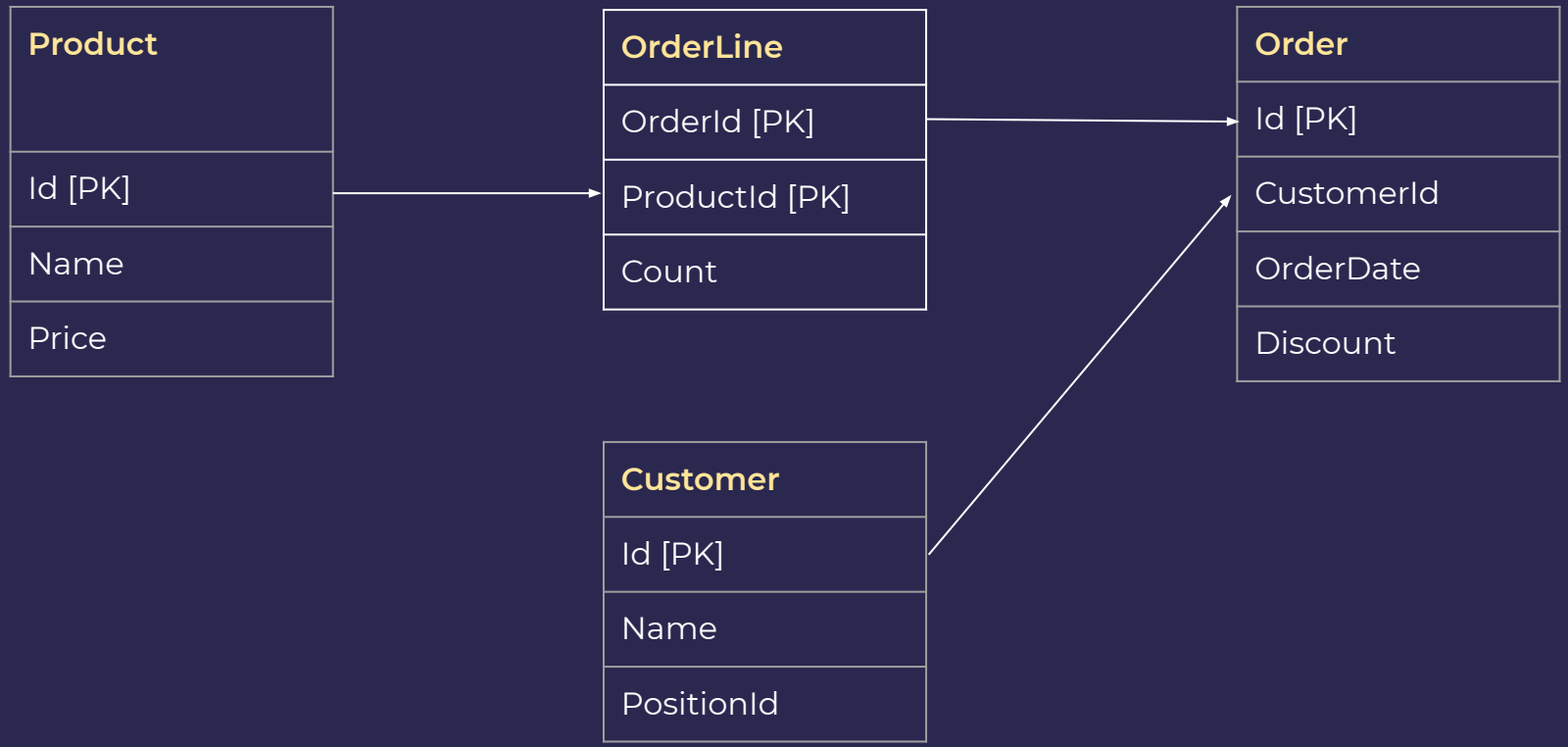
(T-SQL: скалярные и агрегатные функции, агрегирование объединений, группировка)

Артём Трофимушкин

Схема и данные для работы в классе

Для работы в классе мы будем использовать БД, состоящую из четырёх таблиц, хранящих информацию о каталоге и продажах простого магазина:

- Customer : клиент
  - Id : идентификатор
  - Name : имя
- Product : продукт
  - Id : идентификатор
  - Name : наименование
  - Price : цена
- Order : заказ
  - Id : идентификатор
  - CustomerId : идентификатор клиента
  - OrderDate : дата заказа
  - Discount : размер скидки
- OrderLine : позиция заказа
  - OrderId : идентификатор
  - ProductId : идентификатор продукта
  - Count : количество


Product
Id [PK]
Name
Price
OrderLine
OrderId [PK]
ProductId [PK]
Count
Order
Id [PK]
CustomerId
OrderDate
Discount
Customer
Id [PK]
Name
PositionId

Функции T-SQL

В T-SQL существует больше количество встроенных функций. Среди прочих хочется выделить две основные группы:

- Скалярные функции обрабатывают одиночное значение и возвращают одиночное значение
- Агрегатные функции выполняют вычисление на наборе значений и возвращают одиночное значение

Строковые функции в T-SQL

Функция LEN возвращает количество символов строки, исключая конечные пробелы. Может использоваться с константами, при выборке данных или в условном выражении:

```
-- LEN: вызов для константы или переменной
SELECT LEN('Hello, world!');

-- LEN: вызов в качестве вычисляемого поля
SELECT [Field], LEN([StringField]) FROM [Table];

-- LEN: вызов в условном выражении
SELECT * FROM [Table] WHERE LEN([StringField]) > 20;
```

Функции даты/времени

Функция YEAR возвращает год как целое число от даты, переданной аргументом. Может использоваться с константами, при выборке данных или в условном выражении:

```
-- YEAR: вызов для константы или переменной
DECLARE @year AS INT = YEAR(GETUTCDATE());
SELECT @year;

-- YEAR: вызов в качестве вычисляемого поля
SELECT YEAR([DateTimeField]) FROM [Table];

-- YEAR: вызов в условном выражении
SELECT * FROM [Table] WHERE YEAR([DateTimeField]) = @year;
```

Агрегатные/статистические функции

- COUNT, COUNT_BIG : возвращают количество элементов, найденных в группе набора данных. Функции различаются типами возвращаемых значений (int / bigint):

```
SELECT COUNT(*) FROM [Table];
SELECT COUNT(DISTINCT [Field]) FROM [Table];
```

- MAX, MIN : возвращает соответственно максимальное или минимальное значение элементов найденных в группе набора данных:

```
SELECT MAX([Field]), MIN([Field]) FROM [Table];
```

- SUM : возвращают сумму всех, либо только уникальных значений в выражении. Функция SUM может быть использована только для числовых столбцов. Значения NULL пропускаются:

```
SELECT SUM([Field]) FROM [Table];
```

- AVG : возвращает среднее арифметическое группы значений. Значения NULL не учитываются:

```
SELECT AVG([Field]) FROM [Table];
```

Самостоятельная работа #1

Написать запросы, возвращающие следующие данные:

1. Полное количество записей в таблице OrderLine
2. Количество уникальных заказов (по таблице OrderLine)
3. Максимальный номер заказа (по таблице Order)
4. Средний размер скидки (по таблице Order)
5. Дата первой и последней продажи (по таблице Order)
6. Дата последней продажи в 2018 году (по таблице Order)
7. Максимальная длина наименования товара (по таблице Product)

Вложенные запросы в условии

```sql
-- Найти Id и имена клиентов, сделавших заказы в 2018 году
SELECT C.[Id], C.[Name] FROM [Customer] AS C
WHERE C.[Id] IN (
     SELECT O.CustomerId FROM [Order] AS O
     WHERE YEAR(O.OrderDate) = 2018
);

-- Найти Id и название товара с максимальной длиной наименования
-- (надо понимать, что потенциально такой запрос может вернуть N записей,
-- в случае, если у нас окажется N единиц товара, имеющих одинаковую длину,
-- которая окажется максимальной)
SELECT P.[Id], P.[Name] FROM [Product] AS P
WHERE LEN(P.[Name]) = (
     SELECT MAX(LEN(PI.[Name])) FROM [Product] AS PI
);
```

Самостоятельная работа #2

Написать запросы, возвращающие следующие данные:

1. Номер заказа с максимальной скидкой в 2016 году
2. Номер первого заказа в 2019 году
3. Id и имя клиента, получившего максимальную скидку в 2016 году
4. Id и имя клиента, сделавшего первый заказ в 2019 году

Агрегирование значений в объединениях

Алгоритм действий

1. Сначала определяем по схеме, в каких таблицах лежат необходимые для вывода и условий данные
2. Объединяем целевые таблицы, при необходимости используя связь с промежуточными таблицами
3. Определяем условия с помощью инструкции WHERE
4. Перечисляем необходимые поля и/или, при необходимости, вводим расчётные поля

Совместная работа

Найти список товаров с ценой, количеством и стоимостью для заказа с Id = 22, а также посчитать полную стоимость этого заказа

Самостоятельная работа #3

Написать запрос, возвращающий полную итоговую сумму, потраченную Марией (некоторым клиентом)

Группировка значений

Команда GROUP BY позволяет группировать результаты при выборке из базы данных. При этом агрегирующие функции будут применяться к отдельным группам

```
-- Общее количество обработанных заказов
SELECT COUNT(O.[Id])
FROM [Order] AS O;

-- Количество обработанных заказов, сгруппированных по годам
SELECT YEAR(O.[OrderDate]), COUNT(*)
FROM [Order] AS O
GROUP BY YEAR(O.[OrderDate]);
```

Самостоятельная работа #4

1. Написать запрос, возвращающий полную итоговую сумму, потраченную каждым клиентом в формате:
   - Идентификатор клиента
   - Имя клиента
   - Итоговая потраченная сумма
2. Добавить разбивку по годам и сортировку по имени, а зачем по году

Домашняя работа

Спасибо за внимание.