ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ АРХАНГЕЛЬСКОЙ ОБЛАСТИ
«МИРНИНСКИЙ ПРОМЫШЛЕННО-ЭКОНОМИЧЕСКИЙ ТЕХНИКУМ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

ПО ТЕМЕ 3.1 ЯЗЫК SQL

ОП.13в БАЗЫ ДАННЫХ

для специальности: 09.02.01 Компьютерные системы и комплексы

2022 г.

Методические рекомендации для ОП.13в Базы данных разработаны для выполнения практических работ по теме 3.1 Язык SQL и составлены в соответствии с рабочей программой МДК и учебным планом по специальности 09.02.01 «Компьютерные системы и комплексы».

Организация-разработчик: государственное бюджетное профессиональное образовательное учреждение Архангельской области «Мирнинский промышленно-экономический техникум»

Разработчик:
Кузнецова С.П., заведующий дневным отделением техникума

ОДОБРЕНЫ цикловой комиссией дисциплин специальностей 09.02.01 и 13.02.11	Составлены в соответствии с требованиями ФГОС по специальности среднего профессионального образования 09.02.01 «Компьютерные системы и комплексы» и учебным планом
Председатель цикловой комиссии В.И.Письменник	Заместитель директора техникума по учебной работе М.Н.Венедиктова

СОДЕРЖАНИЕ

1 Запрос SQL в MS Access. Создание базы данных «Магазин»

Цель работы:

Учится создавать различные запросы с помощью языка программирования SQL, закрепить знания по теме «Язык запросов SQL»,

Описание таблиц

1) В Microsoft Access создайте базу данных, содержащую шесть таблиц, конструктор которых представлен ниже:

m_category - категории товаров

Имя поля	Тип данных	Описание
id	Счетчик	Код категории товара
title	Текстовый	Название категории

m_income - приход товаров

Имя поля	Тип данных	Описание
id	Счетчик	Код записи
dt	Дата/время	Дата прихода
product_id	Числовой	Код товара
amount	Числовой	Количество прихода
price	Числовой	Цена за единицу

m_outcome - расход товаров

Имя поля	Тип данных	Описание
id	Счетчик	Код записи
dt	Дата/время	Дата продажи
product_id	Числовой	Код товара
amount	Числовой	Количество прихода
price	Числовой	Цена за единицу

m_product - справочник, описание товаров

Имя поля	Тип данных	Описание
id	Счетчик	Код товара
title	Текстовый	Название товара
supplier_id	Числовой	Код поставщика
ctgry_id	Числовой	Код категории товара
unit	Текстовый	Единица измерения
lifedays	Числовой	Срок годности в днях

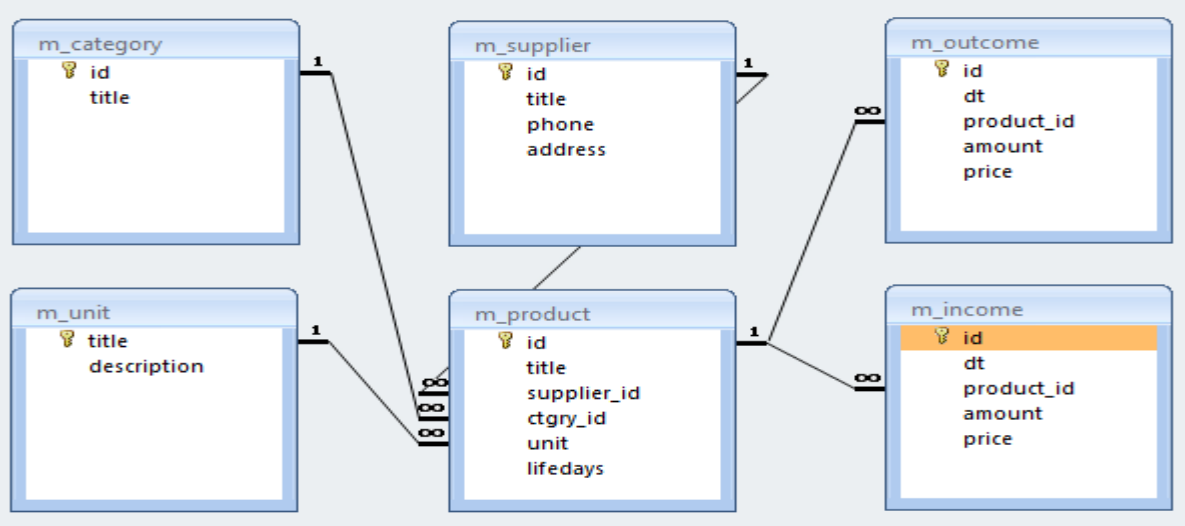
m_supplier - справочник; информация о поставщиках

Имя поля	Тип данных	Описание
id	Счетчик	Код поставщика
title	Текстовый	Имя поставщика
phone	Текстовый	Телефон
address	Текстовый	Адрес

m_unit - справочник; единицы измерения

Имя поля	Тип данных	Описание
title	Текстовый	Тип единицы товара
description	Текстовый	Описание

2) Создайте схему связей таблиц базы данных "Магазин" в соответствии с рисунком:



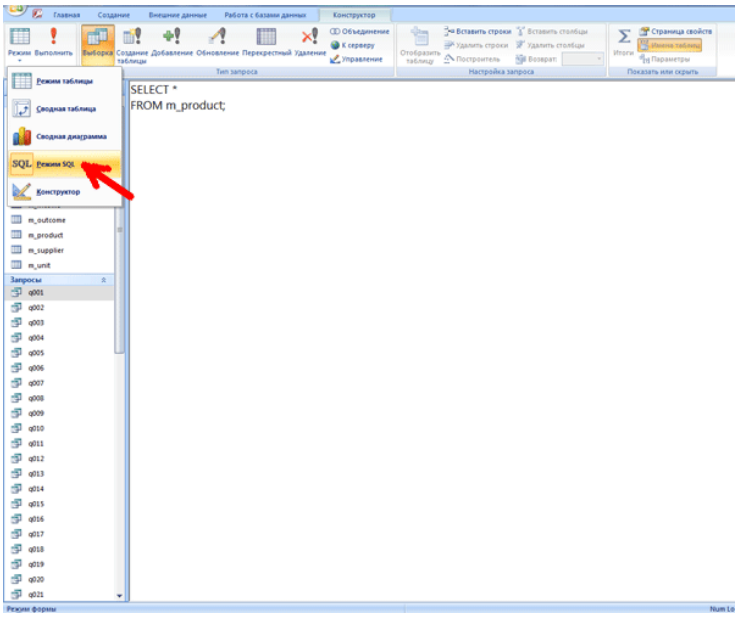
Заполните таблицы соответствующими данными. Для того чтобы увидеть содержимое таблицы, щелкните дважды на названии таблицы на панели слева.

id	title	supplier_i	ctgry_id	unit	lifedays	Добавить по
1	Шоколад плиточный	1	4	шт	90	
2	Конфеты Карамель	3	4	кг	90	
3	Молоко	2	1	л	2	
4	Масло сливочное, упаковка	2	2	шт	30	
5	Масло растительное, бутылка	3	2	шт	90	
6	Масло сливочное, развесное	1	2	кг	10	
7	Мясо говяжье	1	3	кг	5	
8	кефир, упаковка	1	1	шт	2	
9	Хлеб	2	5	шт	1	
10	Батон	3	5	шт	1	
11	Сметана	1	1	шт	2	
(№)						

Для перехода в режим редактирования полей таблицы, на верхней панели выберите режим Конструктора.

2 Простые запросы SQL в MS Access

Для вывода результата запроса SQL, щелкните дважды на названии запроса на панели слева. Для того чтобы перейти в режим редактирования запроса SQL, на верхней панели выберите режим SQL:



В запросе SQL оператор SELECT используется для осуществления выборки из таблиц базы данных.

Запрос SQL Q001. Пример запроса SQL для получения только нужных полей в нужной последовательности:

```
SELECT dt, product_id, amountFROM m_income;
```

Запрос SQL Q002. В этом примере запроса SQL символ звездочки (*) использован для вывода всех столбцов таблицы m_product, иначе говоря, для получения всех полей отношения m_product:

```
SELECT *FROM m_product;
```

Запрос SQL Q003. Инструкция DISTINCT используется для исключения повторяющихся записей и получения множества уникальных записей:

```
SELECT DISTINCT product_idFROM m_income;
```

Запрос SQL Q004. Инструкция ORDER BY используется для сортировки (упорядочивания) записей по значениям определенного поля. Имя поля указывается за инструкцией ORDER BY:

```
SELECT *FROM m_incomeORDER BY price;
```

Запрос SQL Q005. Инструкция ASC используется как дополнение к инструкции ORDER BY и служит для определения сортировки по возрастанию. Инструкция DESC используется как дополнение к инструкции ORDER BY и служит для определения сортировки по убыванию. В случае, когда ни ASC, ни DESC не указаны, подразумевается наличие ASC (default):

```
SELECT *FROM m_incomeORDER BY dt DESC , price;
```

Запрос SQL Q006. Для отбора необходимых записей из таблицы пользуются различными логическими выражениями, которые выражают условие отбора. Логическое выражение приводится после инструкции WHERE. Пример получения из таблицы m_income всех записей, для которых значение amount больше 200:

```
SELECT *
FROM m_income
WHERE amount>200;
```

Запрос SQL Q007. Для выражения сложных условий пользуются логическими операциями AND (конъюнкция), OR (дизъюнкция) и NOT (логическое отрицание). Пример получения из таблицы m_outcome всех записей, для которых значение amount равно 20 и значение price больше или равно 10:

```
SELECT dt, product_id, amount, price
FROM m_outcome
WHERE amount=20 AND price>=10;
```

Запрос SQL Q008. Для объединения данных двух или более таблиц пользуются инструкциями INNER JOIN, LEFT JOIN, RIGHT JOIN. В следующем примере извлекаются поля dt, product_id, amount, price из таблицы m_income и поле title из таблицы m_product. Запись таблицы m_income соединяется с записью таблицы m_product при равенстве значения m_income.product_id значению m_product.id:

```
SELECT dt, product_id, title, amount, price
FROM m_income INNER JOIN m_product
ON m_income.product_id=m_product.id;
```

Запрос SQL Q009. В этом запросе SQL нужно обратить внимание на две вещи: 1) искомый текст заключен в одинарные кавычки ('); 2) дата приведена в формате #Месяц/День/Год#, что верно для MS Access. В других системах формат написания даты может быть другим. Пример вывода информации о поступлении молока 12-го июня 2011 года. Обратите внимание на формат даты #6/12/2011#:

```
SELECT dt, product_id, title, amount, price
FROM m_income INNER JOIN m_product
ON m_income.product_id=m_product.id
WHERE title='Молоко' And dt=#6/12/2011#;
```

Запрос SQL Q010. Инструкция BETWEEN используется для проверки принадлежности некоторому диапазону значений. Пример запроса SQL, выводящий информацию о товарах, поступивших между 1-м и 30-м июнем 2011 года:

```
SELECT *
FROM m_income INNER JOIN m_product
ON m_income.product_id=m_product.id
WHERE dt BETWEEN #6/1/2011# And #6/30/2011#;
```

3 Составные запросы SQL

Один запрос SQL можно вкладывать в другой. Подзапрос - есть не что иное, как запрос внутри запроса. Обычно, подзапрос используется в конструкции WHERE. Но возможны и другие способы использования подзапросов.

Запрос Q011. Выводится информация о товарах из таблицы m_product, коды которых есть и в таблице m_income:

```
SELECT *
FROM m_product
WHERE id IN (SELECT product_id FROM m_income);
```

Запрос Q012. Выводится список товаров из таблицы m_product, кодов которых нет в таблице m_outcome:

```
SELECT *
FROM m_product
WHERE id NOT IN (SELECT product_id FROM m_outcome);
```

Запрос Q013. В этом запросе SQL выводится уникальный список кодов и названий товаров, коды которых есть в таблице m_income, но которых нет в таблице m_outcome:

```
SELECT DISTINCT product_id, title
FROM m_income INNER JOIN m_product
ON m_income.product_id=m_product.id
WHERE product_id NOT IN (SELECT product_id FROM
m_outcome);
```

Запрос Q014. Выводится из таблицы m_category уникальный список категорий, названия которых начинаются на букву М:

```
SELECT DISTINCT title
FROM m_product
WHERE title LIKE 'M*';
```

Запрос Q015. Пример выполнения арифметических операций над полями в запросе и переименования полей в запросе (alias). В этом примере для каждой записи о расходе товара подсчитываются сумма расхода = количество*цена и

размер прибыли, при предположении, что прибыль составляет 7 процентов от суммы продаж:

```
SELECT dt, product_id, amount, price, amount*price AS
outcome_sum,
amount*price/100*7 AS profit
FROM m_outcome;
```

Запрос Q016. Проанализировав и упростив арифметические операции, можно увеличить скорость выполнения запроса:

```
SELECT dt, product_id, amount, price, amount*price AS
outcome_sum,
outcome_sum*0.07 AS profit
FROM m_outcome;
```

Запрос Q017. При помощи инструкции INNER JOIN можно объединить данные нескольких таблиц. В следующем примере, в зависимости от значения ctgry_id, каждой записи таблицы m_income, сопоставляется название категории из таблицы m_category, к которой принадлежит товар:

```
SELECT c.title, b.title, dt, amount, price,
amount*price AS income_sum
FROM (m_income AS a INNER JOIN m_product AS b ON
a.product_id=b.id)
INNER JOIN m_category AS c ON b.ctgry_id=c.id
ORDER BY c.title, b.title;
```

Запрос Q018. Такие функции как SUM - сумма, COUNT - количество, AVG – среднее арифметическое значение, MAX – максимальное значение, MIN – минимальное значение называются агрегатными функциями. Они принимают множество значений, и после их обработки возвращают единственное значение. Пример подсчета суммы произведения полей amount и price при помощи агрегатной функции SUM:

```
SELECT SUM(amount*price) AS Total_Sum
FROM m_income;
```

Запрос Q019. Пример использования нескольких агрегатных функций:

```
SELECT Sum(amount) AS Amount_Sum, AVG(amount) AS Amount_AVG,
MAX(amount) AS Amount_Max, Min(amount) AS Amount_Min,
Count(*) AS Total_Number
FROM m_income;
```

Запрос Q020. В этом примере подсчитана сумма всех товаров с кодом 1, оприходованных в июне 2011 года:

```
SELECT Sum(amount*price) AS income_sum
FROM m_income
WHERE product_id=1 AND dt BETWEEN #6/1/2011# AND #6/30/2011#;.
```

Запрос Q021. Следующий запрос SQL вычисляет на какую сумму было продано товаров, имеющих код 4 или 6:

```
SELECT Sum(amount*price) as outcome_sum
FROM m_outcome
WHERE product_id=4 OR product_id=6;
```

Запрос Q022. Вычисляется на какую сумму было продано 12 июня 2011 года товаров, имеющих код 4 или 6:

```
SELECT Sum(amount*price) AS outcome_sum
FROM m_outcome
WHERE (product_id=4 OR product_id=6) AND dt=#6/12/2011#;
```

Запрос Q023. Задача такова. Вычислить на какую общую сумму было оприходовано товаров категории "Хлебобулочные изделия".

Для решения этой задачи нужно оперировать тремя таблицами: m_income, m_product и m_category, потому что:

- количество и цена оприходованных товаров хранятся в таблице m_income;
- код категории каждого товара хранится в таблице m_product;
- название категории title хранится в таблице m_category.

Для решения данной задачи воспользуемся следующим алгоритмом:

- определение кода категории "Хлебобулочные изделия" из таблицы m_category посредством подзапроса;

- соединение таблиц m_income и m_product для определения категории каждого оприходованного товара;

- вычисление суммы прихода(= количество*цена) для товаров, код категории которых равен коду, определенному вышеуказанным подзапросом.

Итак:

```
SELECT Sum(amount*price) AS income_sum
FROM m_product AS a INNER JOIN m_income AS b ON
a.id=b.product_id
WHERE ctgry_id = (SELECT id FROM m_category WHERE
title='Хлебобулочные изделия');
```

Запрос Q024. Задачу вычисления общей суммы оприходованных товаров категории "Хлебобулочные изделия" решим следующим алгоритмом:

- каждой записи таблицы m_income, в зависимости от значения его product_id, из таблицы m_category, сопоставить название категории;

- выделить записи, для которых категория равна "Хлебобулочные изделия";

- вычислить сумму прихода = количество*цена.

Итак:

```
SELECT Sum(amount*price) AS income_sum
FROM (m_product AS a INNER JOIN m_income AS b ON
a.id=b.product_id)
INNER JOIN m_category AS c ON a.ctgry_id=c.id
WHERE c.title='Хлебобулочные изделия';
```

Запрос Q025. В этом примере вычисляется сколько наименований товаров было израсходовано:

```
SELECT COUNT(product_id) AS product_cnt
FROM (SELECT DISTINCT product_id FROM m_outcome) AS
t;
```

Запрос Q026. Инструкция GROUP BY используется для группировки записей. Обычно записи группируются по значению одного или нескольких полей, и относительно каждой группы применяется какая-либо агрегатная операция. Например, следующий запрос состявляет отчет о продаже товаров. То

есть генерируется таблица, в которой будут названия товаров и сумма, на которую они проданы:

```
SELECT title, SUM(amount*price) AS outcome_sum
FROM m_product AS a INNER JOIN m_outcome AS b
ON a.id=b.product_id
GROUP BY title;
```

Запрос Q027. Отчет о продажах по категориям. То есть генерируется таблица, в которой будут названия категорий товаров, общая сумма, на которую проданы товары данных категорий, и средняя сумма продаж. Функция ROUND использована для округления среднего значения до сотой доли (второй знак после разделителя целой и дробной частей):

```
SELECT c.title, SUM(amount*price) AS outcome_sum,
ROUND(AVG(amount*price),2) AS outcome_sum_avg
FROM (m_product AS a INNER JOIN m_outcome AS b ON
a.id=b.product_id)
INNER JOIN m_category AS c ON a.ctgry_id=c.id
GROUP BY c.title;
```

Запрос Q028. Вычисляется для каждого товара общее и среднее количество его поступлений и выводит информацию о товарах, общее количество поступления которых не менее 500:

```
SELECT product_id, SUM(amount) AS amount_sum,
Round(Avg(amount),2) AS amount_avg
FROM m_income
GROUP BY product_id
HAVING Sum(amount)>=500;
```

Запрос Q029. В этом запросе вычисляется для каждого товара сумма и среднее значение его поступлений, осуществленных во втором квартале 2011 года. Если общая сумма прихода товара не менее 1000, то отображается информация об этом товаре:

```
SELECT title, SUM(amount*price) AS income_sum
```

```
    FROM m_income a INNER JOIN m_product b ON
a.product_id=b.id
    WHERE dt BETWEEN #4/1/2011# AND #6/30/2011#
    GROUP BY title
    HAVING SUM(amount*price)>=1000;
```

Запрос Q030. В некоторых случаях нужно сопоставлять каждой записи некоторой таблицы каждую запись другой таблицы; что называется декартовым произведением. Таблица, образующаяся в результате такого соединения, называется таблицей Декарта. Например, если некоторая таблица A имеет 100 записей и таблица B имеет 15 записей, то их таблица Декарта будет состоять из 100*15=150 записей. Следующий запрос соединяет каждую запись таблицы m_income с каждой записью таблицы m_outcome:

```
    SELECT *
    FROM m_income, m_outcome;
```

Запрос Q031. Пример группирования записей по двум полям. Следующий запрос SQL вычисляет по каждому поставщику сумму и количество поступивщих от него товаров:

```
    SELECT supplier_id, product_id, SUM(amount) AS
amount_sum,
    SUM(amount*price) AS income_sum
    FROM m_income AS a INNER JOIN m_product AS b ON
a.product_id=b.id
    GROUP BY supplier_id, product_id;
```

Запрос Q032. Пример группирования записей по двум полям. Следующий запрос вычисляет для каждого поставщика сумму и количество его продуктов, проданных нами:

```
    SELECT supplier_id, product_id, SUM(amount) AS
amount_sum,
    SUM(amount*price) AS outcome_sum
    FROM m_outcome AS a INNER JOIN m_product AS b ON
a.product_id=b.id
```

```
GROUP BY supplier_id, product_id;
```

Запрос Q033. В этом примере два вышеприведенных запроса (q031 и q032) использованы как подзапросы. Результаты этих запросов методом LEFT JOIN объединены в один отчет. Следующий запрос выводит отчет о количестве и сумме поступивших и реализованных продуктов по каждому поставщику. Следует обратить внимание на то, что если какой-то товар уже поступил, но еще не реализован, то клетка outcome_sum для этой записи будет пустой. Также необходимо отметить, что данный запрос служит только примером использования относительно сложных запросов в качестве подзапроса. Производительность данного запроса SQL при большом объеме данных сомнительна:

```
SELECT *
FROM
(SELECT supplier_id, product_id, SUM(amount) AS
amount_sum,
SUM(amount*price) AS income_sum
FROM m_income AS a INNER JOIN m_product AS b
ON a.product_id=b.id GROUP BY supplier_id, product_id)
AS a
LEFT JOIN
(SELECT supplier_id, product_id, SUM(amount) AS
amount_sum,
SUM(amount*price) AS outcome_sum
FROM m_outcome AS a INNER JOIN m_product AS b
ON a.product_id=b.id GROUP BY supplier_id, product_id)
AS b
ON (a.product_id=b.product_id) AND
(a.supplier_id=b.supplier_id);
```

Запрос Q034. В этом примере два вышеприведенных запроса (q031 и q032) использованы как подзапросы. Результаты этих запросов методом RIGTH JOIN объединены в один отчет. Следующий запрос выводит отчет о сумме платежей каждого клиента по использованным им платежным системам и сумме сделанных

им инвестиций. Следующий запрос выводит отчет о количестве и сумме поступивщих и реализованных продуктов по каждому поставщику. Следует обратить внимание на то, что если какой-то товар уже реализован, но еще не поступил, то клетка income_sum для этой записи будет пустой. Наличие таких пустых клеток является показателем ошибки в учете продаж, так как до продажы сначала необходимо, чтобы соответствующий товар поступил:

```
SELECT *
FROM
(SELECT supplier_id, product_id, SUM(amount) AS
amount_sum,
SUM(amount*price) AS income_sum
FROM m_income AS a INNER JOIN m_product AS b ON
a.product_id=b.id
GROUP BY supplier_id, product_id)  AS a
RIGHT JOIN
(SELECT supplier_id, product_id, SUM(amount) AS
amount_sum,
SUM(amount*price) AS outcome_sum
FROM m_outcome AS a INNER JOIN m_product AS b ON
a.product_id=b.id
GROUP BY supplier_id, product_id)  AS b
ON (a.supplier_id=b.supplier_id) AND
(a.product_id=b.product_id);
```

Запрос Q035. Выводится отчет о сумме доходов и расходов по продуктам. Для этого создается список продуктов по таблицам m_income и m_outcome, затем для каждого продукта из этого списка вычисляется сумма его приходов по таблице m_income и сумма его расходов по таблице m_outcome:

```
SELECT product_id, SUM(in_amount) AS income_amount,
SUM(out_amount) AS outcome_amount
FROM
```

```
    (SELECT product_id, amount AS in_amount, 0 AS
out_amount
    FROM m_income
    UNION ALL
    SELECT product_id, 0 AS in_amount, amount AS
out_amount
    FROM m_outcome)  AS t
    GROUP BY product_id;
```

Запрос Q036. Функция EXISTS возвращает значение TRUE, если переданное ей множество содержит элементы. Функция EXISTS возвращает значение FALSE, если переданное ей множество пустое, то есть не содержит элементов. Следующий запрос выводит коды товаров, которые содержатся как в таблице m_income, так и в таблице m_outcome:

```
SELECT DISTINCT product_id
FROM m_income AS a
WHERE EXISTS(SELECT product_id FROM m_outcome AS b
WHERE b.product_id=a.product_id);
```

Запрос Q037. Выводятся коды товаров, которые содержатся как в таблице m_income, так и в таблице m_outcome:

```
SELECT DISTINCT product_id
FROM m_income AS a
WHERE product_id IN (SELECT product_id FROM m_outcome)
```

Запрос Q038. Выводятся коды товаров, которые содержатся как в таблице m_income, но не содержатся в таблице m_outcome:

```
SELECT DISTINCT product_id
FROM m_income AS a
WHERE NOT EXISTS(SELECT product_id FROM m_outcome AS b
WHERE b.product_id=a.product_id);
```

Запрос Q039. Выводится список товаров, сумма продаж которых максимальная. Алгоритм таков. Для каждого товара вычисляется сумма его продаж. Затем, определяется максимум этих сумм. Затем, для каждого товара

снова вычисляется сумма его продаж, и выводятся код и сумма продаж товаров, сумма продаж которых равна максимальной:

```
SELECT product_id, SUM(amount*price) AS amount_sum
FROM m_outcome
GROUP BY product_id
HAVING SUM(amount*price) = (SELECT MAX(s_amount)
FROM (SELECT SUM(amount*price) AS s_amount FROM
m_outcome GROUP BY product_id));
```

4. Оператор условия в запросах SQL в MS Access

Зарезервированное слово IIF (условный оператор) используется для оценки логического выражения и выполнения того или иного действия в зависимости от результата (TRUE или FALSE).

Запрос Q040. В следующем примере поставка товара считается «малой», если количество меньше 500. В противном случае, то есть количество поступления больше или равно 500, поставка считается «большой»:

```
SELECT dt, product_id, amount,
IIF(amount<500,"малая","большая") AS mark
FROM m_income;
```

Запрос SQL Q041. В случае, когда оператор IIF используется несколько раз, удобнее заменить его оператором SWITCH. Оператор SWITCH (оператор множественного выбора) используется для оценки логического выражения и выполнения того или иного действия в зависимости от результата. В следующем примере поставленная партия считается «малой», если количество товара в партии меньше 500. В противном случае, то есть если количество товара больше или равно 500, партия считается «большой»:

```
SELECT dt, product_id, amount,
SWITCH(amount<500,"малая",amount>=500,"большая") AS
mark
FROM m_income;
```

Запрос Q042. В следующем запросе если количество товара в поступившей партии меньше 300, то партия считается «малой». В противном случае, то есть если условие amount<300 не выполняется, то проверяется является ли количество товаров в партии меньше 500. Если размер партии меньше 500, то она считается «средней». В противном случае партия считается «большой»:

```
SELECT dt, product_id, amount,
IIF(amount<300,"малая",
IIF(amount<1000,"средняя","большая")) AS mark
FROM m_income;
```

5. Группировка в запросах SQL в MS Access

Запрос SQL Q043. В следующем запросе если количество товара в поступившей партии меньше 300, то партия считается «малой». В противном случае, то есть если условие amount<300 не выполняется, то проверяется является ли количество товаров в партии меньше 500. Если размер партии меньше 500, то она считается «средней». В противном случае партия считается «большой»:

```
SELECT dt, product_id, amount,
SWITCH(amount<300,"малая",
amount<1000,"средняя",
amount>=1000,"большая") AS mark
FROM m_income;
```

Запрос SQL Q044. В следующем запросе продажи разделяются на три группы: малые (до 150), средние (от150 до 300), большие (300 и более). Далее, для каждой группы вычисляется итоговая сумма:

```
SELECT Category, SUM(outcome_sum) AS Ctgry_Total
FROM (SELECT amount*price AS outcome_sum,
IIf(amount*price<150,"малая",
IIf(amount*price<300,"средняя","большая")) AS Category
FROM m_outcome)  AS t
GROUP BY Category;
```

Запрос SQL Q045. Функция DateAdd используется для прибавления дней, месяцев или лет к данной дате и получения новой даты. Следующий запрос:

1) к дате из поля dt прибавляет 30 дней и отображает новую дату в поле dt_plus_30d;

2) к дате из поля dt прибавляет 1 месяц и отображает новую дату в поле dt_plus_1m:

```
SELECT dt, dateadd("d",30,dt) AS dt_plus_30d,
dateadd("m",1,dt) AS dt_plus_1m
FROM m_income;
```

Запрос SQL Q046. Функция DateDiff предназначена для вычисления разницы между двумя датами в различных единицах (днях, месяцах или годах). Следующий запрос вычисляет разницу между датой в поле dt и текущей датой в днях, месяцах и годах:

```
SELECT dt, DateDiff("d",dt,Date()) AS last_day,
DateDiff("m",dt,Date()) AS last_months,
DateDiff("yyyy",dt,Date()) AS last_years
FROM m_income;
```

Запрос SQL Q047. Вычисляются количество дней со дня поступления товара (таблица m_income) до текущей даты с помощью функции DateDiff и сопоставляется срок годности (таблица m_product):

```
SELECT a.id, product_id, dt, lifedays,
DateDiff("d",dt,Date()) AS last_days
FROM m_income AS a INNER JOIN m_product AS b
ON a.product_id=b.id;
```

Запрос SQL Q048. Вычисляются количество дней со дня поступления товара до текущей даты, затем проверяется превышает ли это количество срок годности:

```
SELECT a.id, product_id, dt, lifedays,
DateDiff("d",dt,Date()) AS last_days,
IIf(last_days>lifedays,"Да","Нет") AS date_expire
FROM m_income a INNER JOIN m_product b
ON a.product_id=b.id;
```

Запрос SQL Q049. Вычисляются количество месяцев со дня поступления товара до текущей даты. В столбце month_last1 вычисляется абсолютное количество месяцев, в столбце month_last2 вычисляется количество полных месяцев:

```
SELECT dt, DateDiff("m",dt,Date()) AS month_last1,
DateDiff("m",dt,Date())-iif(day(dt)>day(date()),1,0)
AS month_last2
FROM m_income;
```

Запрос SQL Q050. Выводится поквартальный отчет о количестве и сумме оприходованных товаров за 2011 год:

```
SELECT kvartal, SUM(outcome_sum) AS Total
FROM (SELECT amount*price AS outcome_sum, month(dt) AS m,
SWITCH(m<4,1,m<7,2,m<10,3,m>=10,4) AS kvartal
FROM m_income WHERE year(dt)=2011)  AS t
GROUP BY kvartal;
```

Запрос Q051. Следующий запрос помогает выяснить, удалось ли пользователям ввести в систему информацию о расходе товара на сумму большую, чем сумма прихода товара:

```
SELECT product_id, SUM(in_sum) AS income_sum, SUM(out_sum) AS outcome_sum
FROM (SELECT product_id, amount*price as in_sum, 0 as out_sum
from m_income
UNION ALL
SELECT product_id, 0 as in_sum, amount*price as out_sum
from m_outcome)  AS t
GROUP BY product_id
HAVING SUM(in_sum)<SUM(out_sum);
```

Запрос Q052. Нумерацию строк, возвращаемых запросом, реализуют по-разному. Например, можно перенумеровать строки отчета, подготовленного в MS Access, средствами самого MS Access. Перенумеровать можно и с использованием языков программирования, например, VBA или PHP. Однако иногда это необходимо сделать в самом запросе SQL. Итак, следующий запрос пронумерует строки таблицы m_income в соответствии с порядком возрастания значений поля ID:

```
SELECT COUNT(*) as N, b.id, b.product_id, b.amount, b.price
```

```
FROM m_income a INNER JOIN m_income b ON a.id <= b.id
GROUP BY b.id, b.product_id, b.amount, b.price;
```

Запрос Q053. Выводится пятерка лидеров среди продуктов по сумме продаж. Вывод первых пяти записей осуществляется с помощью инструкции TOP:

```
SELECT TOP 5, product_id, sum(amount*price) AS summa
FROM m_outcome
GROUP BY product_id
ORDER BY sum(amount*price) DESC;
```

Запрос Q054. Выводится пятерка лидеров среди продуктов по сумме продаж, и нумерует строки в результате:

```
SELECT COUNT(*) AS N, b.product_id, b.summa
FROM
(SELECT product_id, sum(amount*price) AS summa,
summa*10000000+product_id AS id
FROM m_outcome GROUP BY product_id)  AS a
INNER JOIN
(SELECT product_id, sum(amount*price) AS summa,
summa*10000000+product_id AS id
FROM m_outcome GROUP BY product_id)  AS b
ON a.id>=b.id
GROUP BY b.product_id, b.summa
HAVING COUNT(*)<=5
ORDER BY COUNT(*);
```

Запрос Q055. Следующий SQL-запрос показывает использование математических функций COS, SIN, TAN, SQRT, ^ и ABS в MS Access SQL:

```
SELECT (select count(*) from m_income) as N, 3.1415926
as pi, k,
2*pi*(k-1)/N as x, COS(x) as COS_, SIN(x) as SIN_,
TAN(x) as TAN_,
SQR(x) as SQRT_, x^3 as 'x^3', ABS(x) as ABS_
```

```
    FROM (SELECT COUNT(*) AS k
    FROM m_income AS a INNER JOIN m_income AS b ON
a.id<=b.id
    GROUP BY b.id) t;
```

6. Запросы SQL на обновление в MS Access

Запрос U001. Следующий SQL-запрос на изменение увеличивает на 10% цены на товары с кодом 3 в таблице m_income:

```
UPDATE m_income SET price = price*1.1
WHERE product_id=3;
```

Запрос U002. Следующий SQL-запрос на обновление увеличивает в таблице m_income на 22 единицы количество всех товаров, названия которых начинаются со слова "Масло":

```
UPDATE m_income SET amount = amount+22
WHERE product_id IN (SELECT id FROM m_product WHERE title LIKE "Масло*");
```

Запрос U003. Следующий SQL-запрос на изменение в таблице m_outcome снижает на 2 процента цены на все товары, производителем которых является ООО "Сладкое":

```
UPDATE m_outcome SET price = price*0.98
WHERE product_id IN
(SELECT a.id FROM m_product a INNER JOIN m_supplier b
ON a.supplier_id=b.id WHERE b.title='ООО "Сладкое"');.
```

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Основная литература

1. Фуфаев Э.В. Базы данных: Учебное пособие. – М.: Академия, 2015.
2. Кузин А.В., Демин В.М. Разработка баз данных в системе Microsoft Access: учебник. – М.: ФОРУМ: ИНФРА-М, 2007.
1. Агальцов В.П. Базы данных. – М.: Мир, 2002.
2. Когаловский М.Р. Энциклопедия технологий баз данных. – М.: Финансы и статистика, 2002.
3. Фаронов В.В. Программирование баз данных в Delphi 7. Учебный курс. – СПб.: Питер, 2006.

Интернет-ресурсы

1. Образовательный портал: http\\www.edu.sety.ru
2. Образовательный портал: http\\www.edu.bd.ru