# Описание задания

**Тестовое задание DB** – проектирование БД со скидками.

**Основной список полей/таблиц:**

* % скидки: Наименование, % (может быть дробным)
* Карты: серия карты, номер карты дата и время выпуска карты, дата и время окончания активности карты, дата и время последнего использования, сумма покупок, статус карты (не активирована / активирована / просрочена), % текущая скидка, заказы;
* Заказы(покупки): Номер, дата и время, сумма, % скидки (для конкретного заказа), скидка (расчет)
* Товары: заказ, наименование товара, цена, цена со скидкой

**Необходимо разработать набор хранимых процедур:**

* получение список карт с полями: серия, номер, дата выпуска, дата окончания активности, статус, функцией поиска по полям;
* просмотра профиля карты с историей покупок по ней;
* активация/деактивация карты;
* удаление карты (сперва в корзину с возможностью восстановления);
* генератор карт, с указанием серии и количества генерируемых карт, а срока активности «с-по».
* после истечения срока активности карты, у карты проставляется статус "просрочена";
* записи информации по заказам и товарам для определенной карты.

# Средства разработки

Microsoft SQL Server v18.10 2021

# Проектирование базы данных

На рисунке 1 отображена модель базы данных.

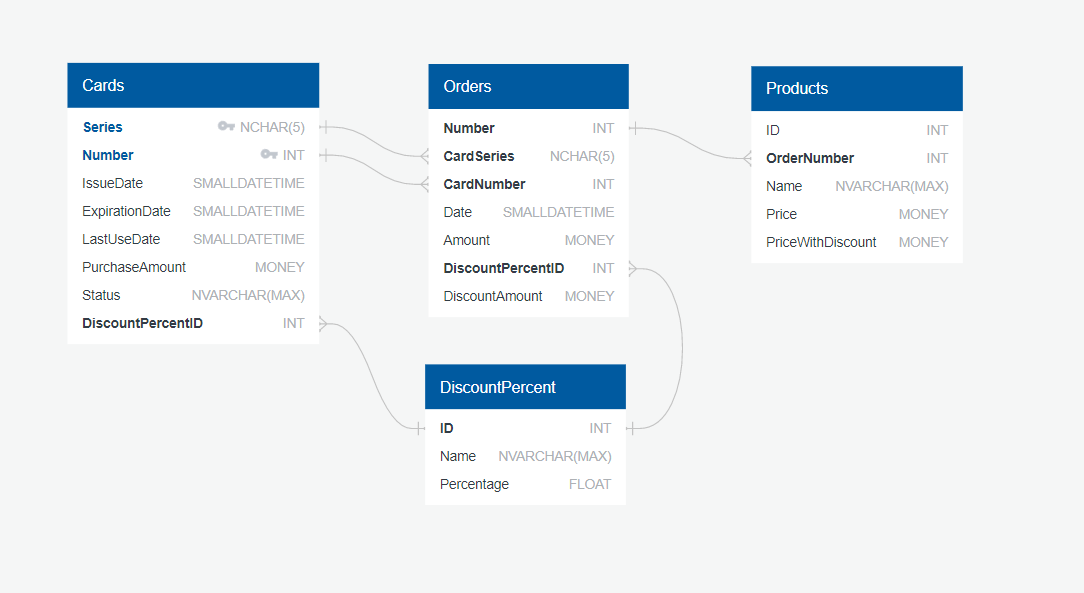


Рисунок 1 – Схема базы данных

Таблица «Cards» отображает структуру таблицу «‎% скидки» из задания, а также содержит суррогатный первичный ключ ID.

Таблица 1 – Структура таблицы «DiscountPercent»

|  |  |  |
| --- | --- | --- |
| Название | Тип, ограничение целостности | Описание |
| ID | int, primary key | Идентификатор скидки |
| Name | nvarchar, not null | Название скидки |
| Percentage | float, not null | Процент скидки |

Таблица «Cards» отображает структуру таблицу «Карты» из задания.

Таблица 2 – Структура таблицы «Cards»

|  |  |  |
| --- | --- | --- |
| Название | Тип, ограничение целостности | Описание |
| Series | nchar(5), primary key | Серия карты |
| Number | int, identity, not null | Номер карты |
| IssueDate | smalldatetime, not null | Дата и время выпуска |
| ExpirationDate | smalldatetime, not null | Дата и время окончания активности карты |
| LastUseDate | smalldatetime, not null | Дата и время последнего использования |
| PurchaseAmount | money, default 0, not null | Сумма покупок |
| Status | nvarchar, check («не активирована», «активирована», «просрочена») | Статус карты |
| DiscountPercent | int, foreign key DiscountPercent(ID) | Текущая скидка |

Таблица «Orders» отображает структуру таблицу «‎Заказы» из задания.

Таблица 3 – Структура таблицы «Orders»

|  |  |  |
| --- | --- | --- |
| Название | Тип, ограничение целостности | Описание |
| Number | int, primary key | Номер заказа |
| CardSeries | nvarchar, foreign key Cards(Series) | Серия карты |
| CardNumber | nchar(5), foreign key Cards(Number) | Номер карты |
| Date | smalldatetime, not null | Дата и время заказа |
| Amount | money, not null | Сумма |
| DiscountPercent | int, foreign key DiscountPercent(ID) | Процент скидки |
| DiscountAmount | money, not null | Скидка (расчет) |

Таблица «Products» отображает структуру таблицу «‎Товары» из задания, а также содержит суррогатный первичный ключ ID.

Таблица 4 – Структура таблицы «Products»

|  |  |  |
| --- | --- | --- |
| Название | Тип, ограничение целостности | Описание |
| ID | int, primary key | Номер товара |
| OrderNumber | int, foreign key Orders(Number) | Номер заказа |
| Name | nvarchar, not null | Наименование товара |
| Price | money, not null | Цена |
| PriceWithDiscount | money, not null | Цена со скидкой |

Создание базы данных и таблиц с необходимыми ограничениями представлен на листинге 1.

USE MASTER;

GO

DROP DATABASE SELLWIN\_TEST;

CREATE DATABASE SELLWIN\_TEST;

GO

USE SELLWIN\_TEST;

GO

CREATE TABLE DiscountPercent

(

ID INT IDENTITY(1,1) CONSTRAINT PK\_DiscountPercentID PRIMARY KEY,

Name NVARCHAR(MAX) NOT NULL,

Percentage FLOAT NOT NULL

);

CREATE TABLE Cards

(

Series NCHAR(5),

Number INT IDENTITY(10000, 1),

IssueDate SMALLDATETIME NOT NULL,

ExpirationDate SMALLDATETIME NOT NULL,

LastUseDate SMALLDATETIME,

PurchaseAmount MONEY DEFAULT 0 NOT NULL,

Status NVARCHAR(MAX) CHECK (Status IN ('не активирована', 'активирована', 'просрочена')),

DiscountPercentID INT CONSTRAINT FK\_Cards\_DiscountPercentID REFERENCES DiscountPercent(ID),

CONSTRAINT PK\_Cards PRIMARY KEY (Series, Number)

);

CREATE TABLE Orders

(

Number INT IDENTITY(1,1) CONSTRAINT PK\_OrdersNumber PRIMARY KEY,

CardSeries NCHAR(5),

CardNumber INT,

Date SMALLDATETIME NOT NULL,

Amount MONEY NOT NULL,

DiscountPercentID INT CONSTRAINT FK\_Orders\_DiscountPercentID REFERENCES DiscountPercent(ID),

DiscountAmount MONEY NOT NULL,

CONSTRAINT FK\_Cards FOREIGN KEY (CardSeries, CardNumber) REFERENCES Cards(Series, Number)

);

--ALTER TABLE Orders ADD DiscountAmount as (Amount \* (SELECT Percentage FROM DiscountPercent WHERE DiscountPercent.ID = DiscountPercentID) / 100) PERSISTED;

CREATE TABLE Products

(

ID INT IDENTITY(1,1) CONSTRAINT PK\_ProductsNumber PRIMARY KEY,

OrderNumber INT CONSTRAINT FK\_OrdersNumber REFERENCES Orders(Number),

Name NVARCHAR(MAX) NOT NULL,

Price MONEY NOT NULL,

PriceWithDiscount MONEY NOT NULL

);

Листинг 1 – Развертывание базы данных

# Создание процедур для работы с данными

* получение список карт с полями: серия, номер, дата выпуска, дата окончания активности, статус, функцией поиска по полям;

В качестве входных параметров процедуры может выступать любая комбинация перечисленных полей, что позволяет осуществить отображение карт по заданным условиям.

CREATE PROCEDURE GetCardList

@series NCHAR(5) = NULL,

@number INT = NULL,

@issue\_date DATE = NULL,

@expiration\_date DATE = NULL,

@status NVARCHAR(MAX) = NULL

AS

BEGIN

BEGIN TRY

SELECT Series, Number, IssueDate, ExpirationDate, Status

FROM Cards

WHERE (@series IS NULL OR Series LIKE '%' + @series + '%')

AND (@number IS NULL OR Number = @number)

AND (@issue\_date IS NULL OR CAST(IssueDate AS DATE) = @issue\_date)

AND (@expiration\_date IS NULL OR CAST(ExpirationDate AS DATE) = @expiration\_date)

AND (@status IS NULL OR Status = @status)

END TRY

BEGIN CATCH

PRINT 'Код ошибки: ' + CAST(ERROR\_NUMBER() as varchar)

PRINT 'Уровень серьезности: ' + CAST(ERROR\_SEVERITY() as varchar)

PRINT 'Текст сообщения: ' + CAST(ERROR\_MESSAGE() as varchar)

END CATCH

END;

Листинг 2 – Процедура получения списка карт

* просмотра профиля карты с историей покупок по ней;

В качестве профиля карты и истории покупок выступают основные поля таблицы Cards, а также номера заказов, связанных с этой картой. Входные параметры процедуры – серия и номер карты.

CREATE PROCEDURE GetCardProfile

@number INT,

@series NCHAR(5)

AS

BEGIN

BEGIN TRY

SELECT Cards.Series, Cards.Number, Cards.IssueDate, Cards.ExpirationDate, Cards.PurchaseAmount, Cards.Status, Cards.DiscountPercentID, Orders.Number

FROM Cards INNER JOIN Orders

ON Cards.Series = Orders.CardSeries AND

Cards.Number = Orders.CardNumber

WHERE (Cards.Series LIKE '%' + @series + '%')

AND (Cards.Number = @number)

END TRY

BEGIN CATCH

PRINT 'Код ошибки: ' + CAST(ERROR\_NUMBER() as varchar)

PRINT 'Уровень серьезности: ' + CAST(ERROR\_SEVERITY() as varchar)

PRINT 'Текст сообщения: ' + CAST(ERROR\_MESSAGE() as varchar)

END CATCH

END;

Листинг 3 – Процедура получения профиля карты

* активация/деактивация карты;

Процедура активации карты заменят значение поля статус карты, переданной в процедуру по номеру и серии, на «активирована».

CREATE PROCEDURE ActivateCard

@number INT,

@series NCHAR(5)

AS

BEGIN

BEGIN TRY

UPDATE Cards SET Status = 'активирована'

WHERE (Cards.Series LIKE '%' + @series + '%')

AND (Cards.Number = @number)

END TRY

BEGIN CATCH

PRINT 'Код ошибки: ' + CAST(ERROR\_NUMBER() as varchar)

PRINT 'Уровень серьезности: ' + CAST(ERROR\_SEVERITY() as varchar)

PRINT 'Текст сообщения: ' + CAST(ERROR\_MESSAGE() as varchar)

END CATCH

END;

Листинг 4 – Процедура активации карты

Процедура деактивации карты работает аналогично и заменят значение на «не активирована».

CREATE PROCEDURE DectivateCard

@number INT,

@series NCHAR(5)

AS

BEGIN

BEGIN TRY

UPDATE Cards SET Status = 'не активирована'

WHERE (Cards.Series LIKE '%' + @series + '%')

AND (Cards.Number = @number)

END TRY

BEGIN CATCH

PRINT 'Код ошибки: ' + CAST(ERROR\_NUMBER() as varchar)

PRINT 'Уровень серьезности: ' + CAST(ERROR\_SEVERITY() as varchar)

PRINT 'Текст сообщения: ' + CAST(ERROR\_MESSAGE() as varchar)

END CATCH

END;

Листинг 5 – Процедура деактивации карты

* генератор карт, с указанием серии и количества генерируемых карт, а срока активности «с-по»;

Процедура генерации создает новые записи в таблице Cards с указанными параметрами, а также идентификатором процента скидки, выступающего в качестве вторичного ключа. Входные параметры – серия, идентификатор процента скидки, дата выпуска, дата окончания срока активности, количество генериуремых карт.

CREATE PROCEDURE GenerateCards

@series NCHAR(5),

@discoutID INT,

@issueDate SMALLDATETIME,

@expirationDate SMALLDATETIME,

@num INT

AS

BEGIN

BEGIN TRY

WHILE @num > 0

BEGIN

INSERT INTO Cards(Series, IssueDate, ExpirationDate, Status, DiscountPercentID)

VALUES (@series, @issueDate, @expirationDate, 'не активирована', @discoutID);

SET @num = @num - 1

END

END TRY

BEGIN CATCH

PRINT 'Код ошибки: ' + CAST(ERROR\_NUMBER() as varchar)

PRINT 'Уровень серьезности: ' + CAST(ERROR\_SEVERITY() as varchar)

PRINT 'Текст сообщения: ' + CAST(ERROR\_MESSAGE() as varchar)

END CATCH

END;

Листинг 6 – Процедура генерации карт

* после истечения срока активности карты, у карты проставляется статус "просрочена";

Процедура истечения срока активности карты заменяет статус карты на «просрочена», если ее дата окончания активности позже, чем текущая дата, полученная с помощью функции GETDATE() и статус не «просрочена», что позволяет обрабатывать меньше строк.

CREATE PROCEDURE SetCardExpired

AS

BEGIN

BEGIN TRY

UPDATE Cards SET Status = 'просрочена'

WHERE ExpirationDate < GETDATE() AND Status <> 'просрочена'

END TRY

BEGIN CATCH

PRINT 'Код ошибки: ' + CAST(ERROR\_NUMBER() as varchar)

PRINT 'Уровень серьезности: ' + CAST(ERROR\_SEVERITY() as varchar)

PRINT 'Текст сообщения: ' + CAST(ERROR\_MESSAGE() as varchar)

END CATCH

END;

Листинг 7 – Процедура истечения срока карты

* записи информации по заказам и товарам для определенной карты;

Процедура записи информации по заказам и товарам для определенной карты обновляет информацию о карте по совершенным с ее помощью заказов. Входные параметры – номер и серия карты.

CREATE PROCEDURE InputToCard

@number INT,

@series NCHAR(5)

AS

BEGIN

BEGIN TRY

UPDATE Cards SET LastUseDate =

(SElECT TOP(1) Orders.Date from Orders

WHERE (Orders.CardSeries LIKE '%' + @series + '%')

AND (Orders.CardNumber = @number)

ORDER BY Orders.Date DESC)

WHERE (Cards.Series LIKE '%' + @series + '%')

AND (Cards.Number = @number)

UPDATE Cards SET PurchaseAmount =

(SELECT SUM(Orders.Amount) FROM ORDERS

WHERE (Orders.CardSeries LIKE '%' + @series + '%')

AND (Orders.CardNumber = @number))

WHERE (Cards.Series LIKE '%' + @series + '%')

AND (Cards.Number = @number)

END TRY

BEGIN CATCH

PRINT 'Код ошибки: ' + CAST(ERROR\_NUMBER() as varchar)

PRINT 'Уровень серьезности: ' + CAST(ERROR\_SEVERITY() as varchar)

PRINT 'Текст сообщения: ' + CAST(ERROR\_MESSAGE() as varchar)

END CATCH

END;

Листинг 8 – Процедура записи информации

* удаление карты (сперва в корзину с возможностью восстановления).

Для удаления карты в корзину было принято решение добавить поле «IsDeleted», принимающего значения «0» и «1», где 1 означает, что карта удалена. При удалении карты значение заменяется на «0», при восстановлении – на «0». Удаление и восстановление карты реализовано двумя процедурами, принимающими в качестве входных параметров серию и номер карты.

ALTER TABLE Cards ADD IsDeleted BIT NOT NULL DEFAULT 0;

GO

CREATE PROCEDURE DeleteCard

@series NCHAR(5),

@number INT

AS

BEGIN

BEGIN TRY

UPDATE Cards Set IsDeleted = 1

WHERE (Cards.Series LIKE '%' + @series + '%')

AND (Cards.Number = @number)

END TRY

BEGIN CATCH

PRINT 'Код ошибки: ' + CAST(ERROR\_NUMBER() as varchar)

PRINT 'Уровень серьезности: ' + CAST(ERROR\_SEVERITY() as varchar)

PRINT 'Текст сообщения: ' + CAST(ERROR\_MESSAGE() as varchar)

END CATCH

END;

GO

CREATE PROCEDURE RestoreCard

@series NCHAR(5),

@number INT

AS

BEGIN

BEGIN TRY

UPDATE Cards Set IsDeleted = 0

WHERE (Cards.Series LIKE '%' + @series + '%')

AND (Cards.Number = @number)

END TRY

BEGIN CATCH

PRINT 'Код ошибки: ' + CAST(ERROR\_NUMBER() as varchar)

PRINT 'Уровень серьезности: ' + CAST(ERROR\_SEVERITY() as varchar)

PRINT 'Текст сообщения: ' + CAST(ERROR\_MESSAGE() as varchar)

END CATCH

END;

Листинг 9 – Процедуры удаления и восстановления карты