Правила форматирования SQL

Чем мы руководствовались при подготовке стилей:

- Простота изучения и поддержки стилей
- Удобочитаемость кода
- Скорость разработки
- Производительность конечного продукта

Ниже даны примеры кода отформатированного по нашим правилам. Код наглядный и дает понимание правил, поэтому сами правила необязательны для чтения и изначально скрыты.

Файлы

- Название определяется согласно стандарту
- Кодировка всех файлов UTF-8

Форматирование

Алиасы

```sql
select s.ID
from sf.Sale as s


update r
set FlagActive = cast(0 as bit)
from dim.Route as r


delete r
from dim.Route as r
where r.FlagActive <> cast(1 as bit)
```

Правила представленные в примере

- Алиас обязателен для объекта и задается с помощью ключевого слова `as`
  - Наименование алиасов определяется согласно стандарту
- При написании `update/delete` запроса, необходимо использовать конструкцию с `from`

Пустые строки

```
begin try
    select a.*
    from geo.Area as a

    -- Пустые строки между логическими блоками
    union all

    select s.*
    from sf.Sale as s
end try
begin catch
    ;throw
end catch

declare
    @ID int
    ,@FlagSuccess int
if @ID = 1
begin
    set @FlagSuccess = 1

    -- Пустая строка перед return
    return
end
```

Правила представленные в примере

- Пустыми строками отделяются разные логические блоки кода

Отступы

```
-- Перед открывающейся скобкой один пробел
if exists (
    -- Комментарий с таким же отступом как и код, к которому он
относится
    select 1
    from geo.Area as a
    where a.Code = 'ЧР'
-- Содержимое скобок переносится на следующую строку
) and (
    select distinct s.ID
    from sf.Sale as s
    where s.Date = getdate()
)
-- На одном уровне с `if` и `begin/end`
begin
    -- Предикат остается на уровне с оператором
    select distinct
        -- Перечисление атрибутов с новой строки
        s.ID_Outlet
        ,a.ID as ID_Area
        ,sum(s.Value) as Value
    from sf.Sale as s
        inner join (
            select a.ID
            from geo.Area as a
            where a.Code = 'ЧР'
        -- Алиас задается без переносов
        ) as a on a.ID = s.ID_Area
    inner join dim.Outlet as o on o.ID = s.ID_Outlet
    group by
        s.ID_Outlet
        ,a.ID
```

```
end
-- На одном уровне с `if` и `begin/end`
else
begin
    select s.*
    from sf.Exclude as e
end

-- При наличии более одного параметра, то все параметры пишутся с новой строки
exec mdt.usp_SafeAddColumn
    @Table = 'rds.Vehicle'
    ,@Column = 'HighLigthColisPick'
    ,@Type = 'bit'
    ,@FlagRequired = 1

-- При наличии только одного параметра, этот параметр пишется на строке выполнения
exec rds.usp_LoadTMS @ID_Record = 100

update o
-- В update скриптах алиасы не пишем
set ID_Shipment = 100
from rds.Order as o
where o.ID = 100

update v
set
    -- Перечисление всех полей с новой строки и одним отступом
    ID_Plan = 100
    ,PlannedShipmentDate = getdate()
from rds.Vehicle as v
    inner join rds.Plan as p on p.ID = v.ID_Plan
```

```
        and p.Name like 'Квартал%'
where v.ID = 100

/*
    Для повышения читаемости кода длинные условия, формулы, выражения и т.п., занимающие более ~75% ширины экрана должны быть разделены на несколько строк. Каждый параметр с новой строки. Применяется выравнивание стеком
*/
select
    so.ID_SKU
    ,coalesce(
        iif(
            (isnull(s.Denominator, 0) * isnull(s.Denominator, 0)) = 0
            ,null
            ,s.Denominator / s.Numerator / s.Denominator
        )
        ,so.VolumeInSU
        ,0
    ) as VolumeInSU
from fact.Sellout as so
    inner join dim.SKU as s on s.ID = so.ID_SKU

-- Вложения выделяются отступами
begin
   set @sObjectName = object_name(@@procid)
   set @sSchema = parsename(@psTemplateName, 2)

   begin
      set @cmd2 = @sPartition

      if @sMessage <> ''
      begin
```

```
            set @sMessage = char(13) + char(10)
        end
    end
end

select distinct d.ID_YearQuarter
from dim.Date as d
-- При использовании `between` условие вместе со словом `and` на новую
строку не переносится
where d.ID_YearMonth between 202201 and 202204
```

Правила представленные в примере

- Отступ пишется через `tab`
- Если выражение или запрос в отдельных скобках не умещаются на одной строке, содержимое скобок начинается с новой строки, с одним отступом
- Все виды `join` пишутся с 1 отступом
    - Если есть `and`, то выравнивать его на 1 табуляцию от `join`
- Закрывающая скобка выносится на отдельную строку, если содержимое занимает несколько строк
- После открывающей и перед закрывающей скобками пробелов нет
- У комментария к строке/блоку отступ такой же, как у строки/блока, к которому написан комментарий
- При наличии нескольких атрибутов, все атрибуты начинаются с новой строки с одним отступом от оператора
    - Предикаты `distinct`, `top` остаются на уровне с оператором
- Отступ перед закрывающей скобкой, стоящей на отдельной строке, равен отступу в строке с открывающей
- `if` и `else` с `begin/end` должны быть на одном уровне
- Алиас при необходимости пишется сразу после закрывающей скобки, на новую строку не переносится
- Отступов перед `set` нет
- Вложенные `begin / end` должны выделяться отступами, `begin / end`, составляющие один блок, должны размещаться на одном уровне
- Любое косметическое выравнивание противоречащее вышестоящим пунктам не приветствуется

Комментарии

```
/*
    Пример многострочного комментария. Как многострочные, так и однострочные комментарии,
    пишутся с соблюдением норм русского языка, понятно и доступно.
    Точка в последнем предложении комментария, в том числе однострочного, не ставится
*/
select a.*
-- Однострочный комментарий
from geo.Area as a

/*
    Оставляем комментарий в шапке объектов
    Расчет данных по вторичным продажам
*/
create or alter function fact.udf_Sellout (@nPartYearMonth)
as
select t.*
from fact.SOP_Sellout as t
where t.nPartYearMonth = @nPartYearMonth

/*
    Оставляем комментарий при написании CTE
    Выбираем месяц, для которого необходимо произвести расчет
*/
with cte_month as (
    select d.ID_YearMonth
    from dim.Date as d
    where d.ID_YearMonth = @nPartYearMonth
)
```

Правила представленные в примере

- Комментарии пишутся с соблюдением всех норм русского языка
- Между `--` и комментарием есть один пробел
- Комментарий пишется непосредственно над строкой кода
- Последнее предложение комментария без точки
- Для комментариев в несколько строк используется конструкция `/* */`
    - Текст комментария начинается с отступа на новой строке
- Комментарии должны быть написаны простым и понятным языком
- В начале скрипта объекта необходимо писать поясняющий комментарий

Пробелы

```
-- После запятой нужен пробел. Между скобкой и содержимым пробел не нужен.
Между скобкой и таблицей пробел нужен
insert into geo.Area (Code, Name)
select
    c.Code
    ,c.Name
from geo.Country as c
-- Перед и после знака равенства нужны пробелы
where c.Code = 'ЧР'

-- Пробел нужен между скобкой и созданием объекта
create table acc.Employee (ID int not null)
-- Пробелов не должно быть при работе с типами полей
alter table acc.Employee add Code varchar(255)

-- Между функцией и открывающей скобкой пробел не нужен
create or alter function rds.udf_UserFlags(@ID_User int)
returns table
as
-- После return скобку не ставим
return
    select
        1 as FlagAdmin
```

```
        ,0 as Flag_RDS_WO
        ,0 as Flag_RDS_FMAN
        ,0 as Flag_RDS_EO
        ,0 as Flag_RDS_PV
        ,0 as Flag_RDS_PR
        ,1 as Priority
    from mdt.Principals(@ID_User) as p
    -- Пробел нужен до знака сравнения и после
    where p.Code = 'Administrators'

select uf.*
-- Пробелов не должно быть при вызове функций
from rds.udf_UserFlags(1) as uf
```

Правила представленные в примере

- Пробел нужен после любой синтаксической единицы, если следом за ней нет синтаксических знаков
- По одному пробелу ставится до и после всех знаков вычисления/сравнения
- При создании объектов нужен пробел после названия объекта
- Пробелов не должно быть при вызове функций
- Пробелов не должно быть при работе с типами полей. Например: `varchar(255)`, `decimal(10, 2)`
- Пробелов не должно быть между скобками и их содержимым
- Пробелов не должно быть в конце строк

Регистр

```
-- Операторы и системные функции пишутся в нижнем регистре
select object_id('sf.Sale')
```

Правила представленные в примере

- Ключевые слова, названия системных функций и все операторы пишутся со строчной буквы

Идентификаторы и переменные

```
declare
    -- Все переменные задаются в одном объявлении
    @ID int
    ,@ID_Area int

-- Дополнительное объявление требуется если нужно использовать ранее объявленные переменные
declare @AreaCode varchar(50) = (select a.Code from geo.Area as a where a.ID = @ID)

select
    -- Префикс udf разделяется знаком `_`
    sf.udf_getNewSaleID(ID) as SaleID
    -- Year - ключевое слово
    ,year(s.MDT_DateCreate) as [Year]
from sf.Sale as s
```

Правила представленные в примере

- Префикс и постфиксы разделяется знаком нижнего подчеркивания `_`
- Все идентификаторы, которые потенциально могут быть приняты за ключевое слово, помещаются в квадратные скобки
- Для объявления переменных `declare` используется один раз. Дальнейшее переменные перечисляются через запятую с новой строки, если явно не требуется писать `declare`

Создание схем и объектов

```
-- Проверяем существование схемы в БД
if not exists (select 1 from sys.schemas where name = 'ft')
begin
       -- Если схемы не существует в БД, то создаем её
       exec('create schema ft');
end
-- Проверяем наличие объекта
```

```sql
if object_id('ft.Brand') is null
begin
	create table ft.Brand (
		-- Обратите внимание, что при create table запятые остаются в конце строк, чтоб не менять код при автоматической генерации
		ID int not null identity,
		Code varchar(255) not null,
		Name varchar(255) not null,
		MDT_ID_PrincipalCreatedBy int not null,
		MDT_DateCreate datetime not null,
		constraint PK_Brand primary key clustered (ID)
	)
	alter table ft.Brand add constraint UK_Brand_Code unique (Code)
	alter table ft.Brand add constraint DF_Brand_MDT_ID_PrincipalCreatedBy_Principal default mdt.ID_User() for MDT_ID_PrincipalCreatedBy
	alter table ft.Brand add constraint FK_Brand_MDT_ID_PrincipalCreatedBy_Principal foreign key (MDT_ID_PrincipalCreatedBy) references mdt.Principal(ID)
	alter table ft.Brand add constraint DF_Brand_MDT_DateCreate default getdate() for MDT_DateCreate
end
-- Создаем (если отсутствует в БД) или изменяем триггер
create or alter trigger dbo.tr_Payroll_insert on dbo.Payroll
after insert
as
begin
	update p
	set ID_status_Status = status.ID_Status('dbo.Payroll', 'dbo.Payroll.CALC')
	from dbo.Payroll as p
	where p.ID_status_Status is null
end
```

Правила представленные в примере

- Для того, чтобы повторное выполнение скрипта не приводило к возникновению ошибки, необходимо проверять существование в БД создаваемого объекта
- Во всех созданных таблицах, кроме `SA` таблиц, обязательно наличие системных полей: `MDT_DateCreate` , `MDT_ID_PrincipalCreatedBy`
- При создании объектов через интерфейс MDT и последующем переносе скрипта в репозиторий, необходимо убедиться, что соблюдены правила форматирования SQL

Конструкции SQL

```
-- Конструкция Case
select case
        when a.Name = 'Чувашская республика'
                -- Дополнительные условия на 2 отступа от when
                and s.ID is not null
            -- Результат на 1 отступ от when
            then 'Продажа'
        when a.Name = 'Московская область'
            then 'Покупка'
        else 'Неизвестно'
    end
from geo.Area as a
    inner join geo.Geography as g on g.ID_Area = a.ID
    left join sf.Sale as s on s.ID_geo_Area = a.ID

-- Конструкция Case в join
select v.ID
from acc.Employee as e
    inner join acc.Vacation as v on v.ID =
            -- Case с новой строки + 1 дополнительный отступ
            case
                when e.field1 = 100
                    then 1
                when e.field1 = 10
```

```
                    then 2
                else e.ID
            end
        and v.field2 =
            case
                when e.field2 = 0
                    then 1
                when e.field2 = 1
                    then 2
                else e.field2
            end
        and v.field3 = e.field3

-- Конструкция if
if exists (
    -- В условных операторах весь блок смещается на 1 отступ
    select s.ID
    from sf.Sale as s
    where s.ID_Sale = @ID_Sale
) and (
    select e.ID
    from sf.Employee as e
    where sf.ID_Employee = @ID_Employee
) and @rn = 1
    and @rank = 1
begin
    select a.*
    from geo.Area as a
        inner join geo.Geography as g on g.ID_Area = a.ID
            and exists (
                select @ID_Geography
            )
    where exists (
```

```
        select 1
    ) and exists (
        select 2
    ) and 1 = 1
end

-- В случае, если if содержит несколько условий, необходимо писать с `begin\end`
if @Count = 1
    and @EmployeeCode = 'ASM'
begin
    set @ID_Employee = -1
end

-- В случае, если после if одна инструкция, допускается написание без `begin\end` с одним отступом
if col_length('rds.Shipment', 'Round') is not null
    alter table rds.Shipment alter column Round varchar(4000)

-- Конструкция merge
merge tpm.DistributorBonusforReview as d
using (
    select
        db.ID_md_CompanyDistributor
        ,db.ID_md_CompanyShipment
        ,db.ID_ref_EmployeeResponsible
        ,db.ID_Review
        ,db.ID_ref_FixPaymentType
        ,-1 as ID_Batch
        ,getdate() as LoadDate
    from cte_FixDistributorBonus as db
) as db on db.ID_Review = d.ID_Review
    and db.ID_md_CompanyDistributor = d.ID_md_Company
```

```
    and db.ID_ref_FixPaymentType = d.ID_tpm_fixPaymentType
when matched then
    update
    set
        ID_md_Company = db.ID_md_CompanyDistributor
        ,ID_md_CompanyShipment = db.ID_md_CompanyShipment
        ,ID_Review = db.ID_Review
        ,ID_Batch = db.ID_Batch
        ,LoadDate = db.LoadDate
when not matched then
    insert (ID_md_Company, ID_md_CompanyShipment, ID_Review, ID_Batch,
LoadDate)
    values   (db.ID_md_CompanyDistributor,   db.ID_md_CompanyShipment,
ID_Review, db.ID_Batch, db.LoadDate);

-- Конструкция values
select
    c.ID as ID_md_CompanyClient
    ,isnull(cs.ID_mapping_MDT, cs.ID) as ID_CompanyShipment
    ,e.ID as ID_ref_Employee
    ,r.ID as ID_Review
    ,s.ValueForJanuary
    ,s.ValueForFebruary
    ,s.ValueForMarch
    ,s.ValueForApril
    ,s.ValueForMay
    ,s.ValueForJune
    ,s.ValueForJuly
    ,s.ValueForAugust
    ,s.ValueForSeptember
from (
    values
        ('1000020021', '2000883532', 0, 0, 0, 0, 0, 0, 0, 0, 0)
```

```
        ,('2000883532', '2000572792', 0, 0, 0, 0, 0, 0, 0, 0, 0)
        ,('1000020022', '2000883532', 0, 0, 0, 0, 0, 0, 0, 0, 0)
        ,('1000020023', '2000883532', 0, 0, 0, 0, 0, 0, 0, 0, 0)
) as s (TFM_ID, Shipment, ValueForJanuary, ValueForFebruary,
ValueForMarch, ValueForApril, ValueForMay, ValueForJune, ValueForJuly,
ValueForAugust, ValueForSeptember)
    inner join md.Company_AdditionalInfo as cai on cai.TFM_ID = s.TFM_ID
    inner join md.Company as c on c.ID = cai.ID_Company
    inner join md.Company as cs on cs.Code = s.Shipment
    inner join mapping.DataSource as ds on ds.ID =
c.ID_mapping_DataSource
        and ds.Code = 'MDT'
    inner join ref.Employee as e on e.ID =
cai.ID_ref_EmployeeResponsible
    inner join tpm.Review as r on r.ID_calendar_Period = 20221
    left join fact.SellOut as so on so.ID_md_Company = c.ID
where so.ID_md_Company is null

insert into fact.Sellout (TFM_ID, Shipment, Volume, Value)
values
    ('1000020021', '2000883532', 10, 100)
    ,('2000883532', '2000572792', 20, 200)
    ,('1000020022', '2000883532', 30, 300)
    ,('1000020023', '2000883532', 40, 400)

if exists (
    -- В условных операторах с одним условием весь блок с условиями
смещается на один отступ
    select 1
)
begin
    select a.Region
    from geo.Area as a
```

```
    union all

    select s.Region
    from geo.Subject as s
end

-- Соединения SQL
select
    fso.nPartYearMonth
    ,d.Date
    ,ph.ID_SKU
    ,fso.Volume
    ,fso.Value
from fact.SOP_Offtake as fso
    inner join dim.ProductHierarchy as ph on ph.ID = fso.ID_ProductHierarchy
        and ph.FlagExcludeSegment = cast(0 as bit)
        and ph.ProductBrandFlagConcurrent = cast(0 as bit)
    -- Сперва указываем поле присоединяемой таблицы
    left join dim.Date as d on d.ID = fso.ID_Date
```

Правила представленные в примере

- Все виды `join` указываются явно
- При соединение двух таблиц, сперва после `on` указываем поле присоединяемой таблицы
- При написании конструкции с `case`, необходимо, чтобы `when` был под `case` с 1 отступом, `then` с 2 отступами, дополнительные условия `and` с 3 отступами, для визуального понимания написанных условий

Лучшие Практики

- Рекомендуется вместо подзапроса использовать CTE (Common Table Expression), временную таблицу или табличную переменную
    - Пример:

```sql
-- Собираем данные по дням
with cte_date as (
    select
        mt.bdate
        ,mt.ID_Date
    from dbo.Date as mt
)
select
    e.Name as EmployeeName
    ,d.ID_Date
from dbo.Employee as e
    inner join cte_date as d on d.bdate = e.bdate

-- Если есть необходимость ставим `;` в начале строки перед
`with`
-- Для удобство переноса кода в запросе
declare @DateBegin date = '2020-01-01'

;with cte_vac (
    select vac.ID_Employee
    from Vacation as vac
    where vac.DateBegin between @DateBegin and '2020-12-31'
        or vac.DateEnd between '2020-01-01' and '2020-12-31'
)
select
    emp.Code
    ,emp.Name
from Employee as emp
where emp.ID not in (
    select v.ID_Employee
    from cte_vac as v
)
```

- Рекомендуется создавать `identity ID` поля только для следующих таблиц:
  - Для справочников — `int`
  - Для небольших справочников — `tinyint`
- Рекомендуется в случае необходимости создавать фиктивную партицию в таблицах фактов со значением `190001`
- Рекомендуется при объявлении типов не использовать длину поля `max`
  - Пример:
    - `varchar(8000)`
    - `nvarchar(4000)`
- Рекомендуется избегать неявных преобразований в типах данных
  - Пример:

```
select
    ps.ID as ID_ProductSKU
    ,ps.ID_Brand
from dbo.D_ProductSKU as ps
-- Преобразуем значение в bit, так как `1` значение типа int
where ps.FlagConcurrentBrand = cast(1 as bit)
    -- Преобразуем значение в `int`, так как UID_DS `varchar`
    and try_cast(ps.UID_DS as int) = 101121
```

- Рекомендуется писать проверки только для DDL инструкций
  - Пример:

```
if col_length('rds.Shipment', 'Round') is not null
    alter table rds.Shipment alter column Round varchar(4000)
```

Оптимизация

- Повторяющийся код должен быть вынесен в отдельную структуру для повторного использования
- Стараться избегать применения `cross apply` , `outer apply`
- Стараться избегать применения `cursor` , по возможности применять вместо него `while`