**Query 1**: 判断一张表的字段数据是否在另外一张表可以用left join 和where (不用in, not in )

select Name as Customers from Customers a left join orders b on a.Id = b.CustomerId

where b.Id is null

**Query 2**: 删除表中的重复数据（有ID字段）, 按照字段分组找出需要保留的ID

delete from emails where id not in (select min(id) from emails group by email, name)

**Query 3:** 返回表中第二高的薪资（没有则返回null）

select

case when max(salary) is not null then max(salary) else null end as SecondHighestSalary

from employee where salary < (select max(salary) from employee)

**Query 4:** 返回表中第n高的薪资(for example 5)

select top 1 salary

from (select distinct top 5 salary from worker order by salary desc) a order by salary asc

**Query 5:** 返回表中第n低的薪资(for example 5)

select top 1 salary

from (select distinct top 5 salary from worker order by salary asc) a order by salary desc

**Query 6:** 返回表中第n高的薪资(for example 5), 不用top

select a.salary from worker a where 5 = (

select count(distinct b.salary) from salary b where a.salary <= b.salary)

**Query 7:** 返回表中第n低的薪资(for example 5), 不用top

select a.salary from worker a where 5 = (

select count(distinct b.salary) from salary b where a.salary >= b.salary)

**Query 8:** 返回表中每个部门薪资最高的员工名字

with base as (

select department, max(salary) from worker group by department

)

select \* from worker w join base b

where w.department = b.department and w.salary = b.salary

**Query 9:** 返回表中每个部门薪资第二高的员工名字

with base as (

SELECT DEPARTMENT , max(SALARY ) as SALARY

FROM Worker group by DEPARTMENT

), secbase as (

select sectable.department, max(sectable.salary) as salary

from

(select w.\* from worker w join base b on w.DEPARTMENT = b.department

where w.SALARY < b.salary ) sectable

group by sectable.department

)

select \* from worker w join secbase s on w.DEPARTMENT = s.department and w.SALARY = s.salary

**方法2：**

select \* from worker a

where 2 = (

select count(distinct salary) from worker b

where a.salary <= b.salary

and a.department = b.department

)

order by a.department , a.SALARY desc

select b.Name as Department , a.Name as Employee , a.Salary

from Employee a , Department b

where 3 >= (SELECT count(distinct Salary) from Employee b WHERE a.Salary <= b.Salary and a.DepartmentId = b.DepartmentId )

and a.DepartmentId = b.Id

order by b.Name , a.Salary desc

**Query 10:** 返回表中前50%的数据记录

with base as (

SELECT \*, row\_number()over(order by getdate()) as rownum

FROM Worker

)

select \* from base where rownum <= (select count(rownum)/2 from base )

**Query 11:** 返回表中最后n条数据记录, for example 5

with base as (

SELECT \*, row\_number()over(order by getdate()) as rownum

FROM Worker

)

select \* from base where rownum > (select max(rownum) - 5 from base )

**Query 12:** Write an SQL query to fetch duplicate records having matching data in some fields of a table.

select WORKER\_TITLE, AFFECTED\_FROM, count(\*)

from title

group by WORKER\_TITLE, AFFECTED\_FROM

having count(\*) > 1

**Query 13:** 连接字段函数concat()

select concat(FIRST\_NAME, ' ', LAST\_NAME) as worker\_name, salary from worker

**Query 14:** 时间函数year(), month() JOINING\_DATE = ‘2014-02-20 09:00:00’

select \* from worker where year(JOINING\_DATE) = 2014 and month(JOINING\_DATE) = 2

**Query 15:** SQL query转义字符(sql server) ESCAPE （通配符wildcard % \_ ）

select \* from worker where DEPARTMENT like '/\_R' ESCAPE '/'

**Query 16:** 字符串函数replace(), len(), ltrim(), rtrim(), charindex(), substring(), upper(), lower()

select replace(FIRST\_NAME, 'a', 'A') from worker

select len(FIRST\_NAME) from worker

select ltrim(FIRST\_NAME), rtrim(FIRST\_NAME) from worker

select charindex('a', first\_name **COLLATE Latin1\_General\_CS\_AS**) from worker #区分大小写

select charindex('a', first\_name) from worker #不区分大小写

select substring(first\_name, 1,3) from worker #index从1开始

select upper(first\_name), lower(first\_name) from worker #全部大写或小写

**Query 17:** 排序函数dense\_rank() #序号连续, rank() #序号不连续

SELECT

Score,

DENSE\_RANK() OVER (ORDER BY Score DESC) AS [Rank],

rank() over (ORDER BY Score DESC) as [RACK2]

FROM Scores

**Query 18:** 创建function的语法：

CREATE FUNCTION getNthHighestSalary(@N INT) RETURNS INT AS

BEGIN

RETURN (

/\* Write your T-SQL query statement below. \*/

select distinct b.salary

from (select salary, dense\_rank() over (order by salary desc) as rank from Employee) b

where b.rank = @N

);

END

**Query 19:** 创建存储过程：

if (exists (select \* from sys.objects where name = 'getBookId'))

drop proc getBookId

go

create proc getBookId(

@bookAuth varchar(**20**),--输入参数,无默认值

@bookId int output --输入/输出参数 无默认值

)

as

select @bookId=book\_id from books where book\_auth=@bookAuth

--执行getBookId这个带返回值的存储过程

declare @id int --声明一个变量用来接收执行存储过程后的返回值

exec getBookId '孔子',@id output

select @id as bookId;--as是给返回的列值起一个名字

**Query 20**: with … as 子句计算取消率

WITH total\_number\_of\_requests AS (

SELECT request\_at, COUNT(Id) AS 'Requests'

FROM(

SELECT DISTINCT Id,Request\_at

FROM Trips AS A

INNER JOIN Users AS B ON A.Client\_Id = B.Users\_Id

INNER JOIN Users AS C ON A.Driver\_Id = C.Users\_Id

WHERE B.Banned = 'No' AND C.Banned = 'No'

AND A.Request\_at BETWEEN '2013-10-01' AND '2013-10-03'

)Z

GROUP BY Request\_at

), total\_number\_of\_cancels AS (

SELECT request\_at, COUNT(Cancels) AS 'Cancels'

FROM(

SELECT request\_at,Id AS 'Cancels'

FROM Trips AS A

INNER JOIN Users AS B ON A.Client\_Id = B.Users\_Id

INNER JOIN Users AS C ON A.Driver\_Id = C.Users\_Id

WHERE B.Banned = 'No' AND C.Banned = 'No'

AND A.Request\_at BETWEEN '2013-10-01' AND '2013-10-03'

AND A.Status IN ('cancelled\_by\_driver','cancelled\_by\_client')

) Z

GROUP BY request\_at

)

SELECT A.request\_at AS Day,

ROUND(CAST(ISNULL(B.Cancels,0.0) AS FLOAT)/A.Requests,2) AS 'Cancellation Rate'

FROM total\_number\_of\_requests AS A

LEFT JOIN total\_number\_of\_cancels AS B ON A.request\_at=B.request\_at

ROUND函数：按照指定的位数进行四舍五入。round(number, 2) #小数点后两位

CAST函数: 类型转换函数, CAST … AS FLOAT 转成浮点类

**Query 21**: 连续两天的数据进行比较， DATEADD()

select w.id

from weather w

join weather w1 on w.RecordDate = DATEADD(day, 1, w1.RecordDate)

and w.Temperature > w1.Temperature

**Query 22**: 数据大于某值并且ID或其他字段值连续三条或三条以上的数据查询

WITH base AS (

SELECT id,visit\_date, people, ROW\_NUMBER()OVER(ORDER BY visit\_date) AS 'R'

FROM stadium WHERE people >= 100

), sequence AS (

SELECT A.id FROM base AS A

INNER JOIN base AS B ON A.id =B.id-1 AND A.R=B.R-1

INNER JOIN base AS C ON A.id =C.id-2 AND A.R=C.R-2 )

SELECT A.id,visit\_date, people FROM stadium AS A WHERE A.id IN (SELECT id FROM sequence) UNION

SELECT A.id,visit\_date, people FROM stadium AS A WHERE A.id IN (SELECT id+1 FROM sequence) UNION

SELECT A.id,visit\_date, people FROM stadium AS A WHERE A.id IN (SELECT id+2 FROM sequence)

**Query 23:** 表中数据的值进行奇偶行互换方法：

select \* from (

select

a.id , case when b.student is null then a.student else b.student end as student

from seat a

left join seat b on a.id = b.id - 1

where a.id % 2 <> 0

union

select

a.id , case when b.student is null then a.student else b.student end as student

from seat a

left join seat b on a.id = b.id + 1

where a.id % 2 = 0) A

order by id

**Query 24:** case when … then … when … then … end

update salary

set sex = (case when sex = 'm' then 'f' when sex = 'f' then 'm' end)

**Query 25:** 纵表转成横表的方法（ID对应的每个月份都有数据，转为横表显示）

select id, sum(case when month = 'Jan' then revenue else NULL end) as Jan\_Revenue ,

sum(case when month = 'Feb' then revenue else NULL end) as Feb\_Revenue ,

sum(case when month = 'Mar' then revenue else NULL end) as Mar\_Revenue ,

sum(case when month = 'Apr' then revenue else NULL end) as Apr\_Revenue ,

sum(case when month = 'May' then revenue else NULL end) as May\_Revenue ,

sum(case when month = 'Jun' then revenue else NULL end) as Jun\_Revenue ,

sum(case when month = 'Jul' then revenue else NULL end) as Jul\_Revenue ,

sum(case when month = 'Aug' then revenue else NULL end) as Aug\_Revenue ,

sum(case when month = 'Sep' then revenue else NULL end) as Sep\_Revenue ,

sum(case when month = 'Oct' then revenue else NULL end) as Oct\_Revenue ,

sum(case when month = 'Nov' then revenue else NULL end) as Nov\_Revenue ,

sum(case when month = 'Dec' then revenue else NULL end) as Dec\_Revenue

from Department

group by id

**Query 26:** 创建空表

select \* into dupblicate\_a from worker where 1=2

**Query 27: 存储过程例子：**

IF EXISTS ( SELECT name FROM sysobjects WHERE type = 'P' AND uid = user\_id()

AND name = 'mkoid' )

DROP PROC mkoid

go

CREATE PROC mkoid (

@oid id\_TY OUTPUT

)

AS BEGIN

DECLARE @prefix binary(3), @chkfix tinyint

DECLARE @seqnum binary(4), @chksum tinyint

SELECT @oid = NULL -- in case it breaks!

SET transaction isolation level 1

SELECT @prefix = prefix, @chkfix = chksum

FROM mkoid\_prefix

IF ( @@error != 0 ) RETURN 1

WHILE @chksum IS NULL OR @chksum = 0

BEGIN

INSERT mkoid\_seqnum VALUES ()

IF ( @@error != 0 ) RETURN 2

SELECT @seqnum = convert(binary(4), convert(int,@@identity))

SELECT @chksum = @chkfix ^ substring(@seqnum, 1, 1)

SELECT @chksum = @chksum ^ substring(@seqnum, 2, 1)

SELECT @chksum = @chksum ^ substring(@seqnum, 3, 1)

SELECT @chksum = @chksum ^ substring(@seqnum, 4, 1)

END

SELECT @oid = @prefix + @seqnum + convert(binary(1), @chksum)

RETURN 0

END

go