**Use Cases for SSMS / MySQL**

1. For every schema, list Schema, # tables, # records

Create or alter procedure SchemaInformation

as

select s.SCHEMA\_NAME, count(distinct t.table\_id) as #Table, count(a.Attributes\_ID) #Records

from DG\_Attributes a

inner join DG\_Tables t on t.Table\_ID = a.Table\_id

inner join DG\_Schema s on s.schema\_ID = t.Database\_ID

group by s.schema\_name

go

exec SchemaInformation

1. For every schema, list table name, # attributes,# PK\_ attr, # FK\_ attr, # NN\_attr, # IDX\_attr, # CC\_attr, # UQ\_attrs, # PGM\_attr

create procedure list\_table\_ALLINFO as

select schema\_name, t.table\_name, count(a.attributes\_Id) as NumberOfAttributes,

count(p.pk\_id) as NoOfPKs, count(f.fk\_id) NoOfFKs, count(c.cc\_id) NoOfCCs, count(i.idx\_id) NoOfIDXs,

count(n.nn\_id) NoOfNNs, count(pg.pgm\_id) NoOfPGMs, count(u.uqi\_id) NoOfUQIs

from DG\_Attributes a

left outer join DG\_Tables t on a.Table\_id = t.Table\_ID

left outer join DG\_Attributes\_ConstCheck c on c.Attributes\_ID = a.Attributes\_ID

left outer join DG\_Attributes\_FK f on f.Attributes\_ID = a.Attributes\_ID

left outer join DG\_Attributes\_Index i on i.Attributes\_ID = a.Attributes\_ID

left outer join DG\_Attributes\_NotNull n on n.Attributes\_ID = a.Attributes\_ID

left outer join DG\_Attributes\_PK p on p.Attributes\_ID = a.Attributes\_ID

left outer join DG\_Attributes\_Program pg on pg.Attributes\_ID = a.Attributes\_ID

left outer join DG\_Attributes\_Unique u on u.Attributes\_ID = a.Attributes\_ID

join DG\_Schema s on t.Database\_ID = s.schema\_ID

group by schema\_name, t.Table\_Name

go

1. For every schema, list Table\_ID, Table\_Name, attribute\_id, attribute\_Name, PK\_ID, FK\_ID, NN\_ID, constraint\_type, and constraint\_Name

Zelalem Denekew

8 days ago

– For DGDB

create or alter procedure Schema\_Table\_Information

as

select s.schema\_name, t.table\_id, t.table\_name, a.attributes\_id, a.attributes\_name,

p.pk\_id, p.constraint\_name, f.fk\_id, f.constraint\_name, n.nn\_id, n.constraint\_name,

c.cc\_id, c.constraint\_name

from DG\_attributes a

inner join DG\_Tables t on t.Table\_ID = a.Table\_id

inner join DG\_Schema s on s.schema\_ID = t.Database\_ID

left outer join DG\_Attributes\_PK p on p.Attributes\_ID = a.Attributes\_ID

left outer join DG\_Attributes\_FK f on f.Attributes\_ID = a.Attributes\_ID

left outer join DG\_Attributes\_ConstCheck c on c.Attributes\_ID = a.Attributes\_ID

left outer join DG\_Attributes\_NotNull n on n.Attributes\_ID = a.Attributes\_ID

group by s.schema\_name, t.Table\_ID, t.Table\_Name, a.attributes\_id, a.attributes\_name,

p.pk\_id, p.constraint\_name, f.fk\_id, f.constraint\_name, n.nn\_id, n.constraint\_name,

c.cc\_id, c.constraint\_name

go

execute Schema\_Table\_Information

1. For every schema, list tables with PK but without FK and IDX

create or alter procedure all\_pk\_without\_fk\_IDX

as

select distinct schema\_name, t.table\_name, Attributes\_Name, p.PK\_ID

from DG\_Attributes a

inner join DG\_Tables t on t.Table\_ID = a.Table\_id

left outer join DG\_Attributes\_PK p on p.Attributes\_ID = a.Attributes\_ID

left outer join (select Attributes\_ID from DG\_Attributes\_FK) f on f.Attributes\_ID = a.Attributes\_ID

left outer join (select Attributes\_ID from DG\_Attributes\_Index) i on i.Attributes\_ID = a.Attributes\_ID

join DG\_Schema s on t.Database\_ID = s.schema\_ID

where f.Attributes\_ID is null

and i.Attributes\_ID is null and p.Attributes\_ID is not null

go

1. For every database/schema, rank them according to number of NN (Not Null) have created

create or alter procedure RankOfSchemaOnNN

as

select s.schema\_name, rank () over (order by count (n.nn\_id)) NN\_Rank, count(n.nn\_id) CountNN

from DG\_Attributes a

inner join DG\_Tables t on t.Table\_ID = a.Table\_id

inner join DG\_Schema s on s.schema\_ID = t.Database\_ID

left outer join DG\_Attributes\_NotNull n on n.Attributes\_ID = a.Attributes\_ID

group by s.schema\_name

go

exec RankOfSchemaOnNN

1. For every database, list tables without PK

create or alter procedure all\_table\_no\_pk

as

select schema\_name, Table\_Name

from DG\_Tables t join DG\_Schema s on t.Database\_ID = s.schema\_ID

where t.Table\_Name not in (

select distinct t.table\_name

from DG\_Attributes a

inner join DG\_Tables t on t.Table\_ID = a.Table\_id

left outer join DG\_Attributes\_PK p on p.Attributes\_ID = a.Attributes\_ID

left outer join (select Attributes\_ID from DG\_Attributes\_FK) f on f.Attributes\_ID = a.Attributes\_ID

left outer join (select Attributes\_ID from DG\_Attributes\_Index) i on i.Attributes\_ID = a.Attributes\_ID

join DG\_Schema s on t.Database\_ID = s.schema\_ID

where p.Attributes\_ID is null

)

Go

1. For every database, list tables without IDX

create or alter procedure all\_table\_no\_idx

as

select schema\_name, Table\_Name

from DG\_Tables t join DG\_Schema s on t.Database\_ID = s.schema\_ID

where t.Table\_Name not in (

select distinct t.table\_name

from DG\_Attributes a

inner join DG\_Tables t on t.Table\_ID = a.Table\_id

left outer join DG\_Attributes\_PK p on p.Attributes\_ID = a.Attributes\_ID

left outer join (select Attributes\_ID from DG\_Attributes\_FK) f on f.Attributes\_ID = a.Attributes\_ID

left outer join (select Attributes\_ID from DG\_Attributes\_Index) i on i.Attributes\_ID = a.Attributes\_ID

join DG\_Schema s on t.Database\_ID = s.schema\_ID

where i.Attributes\_ID is null

)

Go

1. For every database, list tables with PK but without FK

create procedure all\_pk\_without\_fk as

select distinct schema\_name, t.table\_name, Attributes\_Name, p.PK\_ID, f.Attributes\_ID as FK\_ID, i.Attributes\_ID as IDX\_ID

from DG\_Attributes a

inner join DG\_Tables t on t.Table\_ID = a.Table\_id

left outer join DG\_Attributes\_PK p on p.Attributes\_ID = a.Attributes\_ID

left outer join (select Attributes\_ID from DG\_Attributes\_FK) f on f.Attributes\_ID = a.Attributes\_ID

left outer join (select Attributes\_ID from DG\_Attributes\_Index) i on i.Attributes\_ID = a.Attributes\_ID

join DG\_Schema s on t.Database\_ID = s.schema\_ID

where f.Attributes\_ID is null and p.Attributes\_ID is not null

go

1. For every database/schema, list tables with PK but without IDX

create procedure all\_pk\_no\_idx as

Select schema\_name, t.Table\_Name, a.Attributes\_Name, p.PK\_ID,u.IDX\_ID

From DG\_Attributes a

inner join DG\_Attributes\_PK p on a.Attributes\_ID=p.Attributes\_ID

left outer join DG\_Attributes\_Index u on a.Attributes\_ID=u.Attributes\_ID

join DG\_Tables t on a.Table\_id = t.Table\_ID

join DG\_Schema s on t.Database\_ID = s.schema\_ID

where u.Attributes\_ID is null

order by a.Table\_id;

go

1. For every database/schema, list tables with CC

Create procedure CC\_tables as

SELECT s.schema\_name, t.Table\_Name, a.Attributes\_Name, c.CC\_ID

FROM DG\_Attributes a

inner join DG\_Attributes\_ConstCheck c on a.Attributes\_ID = c.Attributes\_ID

join DG\_Tables t on a.Table\_id = t.Table\_ID

join DG\_Schema s on t.Database\_ID = s.schema\_ID

Go

1. For every database/schema, list tables with PGM

Create procedure PGM\_tables as

SELECT s.schema\_name, t.Table\_Name, a.Attributes\_Name, p.PGM\_ID

FROM DG\_Attributes a

inner join DG\_Attributes\_Program p on a.Attributes\_ID = p.Attributes\_ID

join DG\_Tables t on a.Table\_id = t.Table\_ID

join DG\_Schema s on t.Database\_ID = s.schema\_ID

Go

1. For every database/schema, list all views (View name, View description)

Create procedure View\_tables as

SELECT s.schema\_name, t.Table\_Name, v.View\_ID, i.V\_name, i.description

FROM DG\_Tables t

inner join DG\_View\_Table v on t.Table\_ID = v.Table\_ID

join DG\_Schema s on t.Database\_ID = s.schema\_ID

inner join DG\_Views i on v.View\_ID=i.View\_ID

Go

1. For every database/schema, list all table with audit table

create or alter procedure TablesWithAuditTable

as

select s.schema\_name, t.table\_name

from DG\_Tables t

inner join DG\_Schema s on s.schema\_ID = t.Database\_ID

inner join DG\_Audit\_Table a on a.Table\_ID = t.Table\_ID

group by s.schema\_name, t.Table\_Name

go

exec TablesWithAuditTable

1. For every database/schema, list schema, users, role, Dept

create or alter procedure UserInformation

as

select s.schema\_name, u.user\_id as users, r.role\_name as roles, un.unit\_name as Dept

from DG\_User\_Database d

inner join DG\_Schema s on s.schema\_ID = d.Database\_ID

inner join DG\_Users u on u.User\_ID = d.User\_ID

inner join DG\_User\_Unit\_Role ur on ur.User\_ID = u.User\_ID

inner join DG\_Role r on r.Role\_ID = ur.Role\_ID

inner join DG\_Unit un on un.Unit\_ID = ur.Unit\_ID

group by s.schema\_name, u.User\_ID, r.Role\_Name, un.Unit\_Name

go

exec UserInformation

1. For every database/schema, list schema equipped with triggers to manage change in users

create or alter procedure SchemaTableTriggers as

select s.schema\_name, t.table\_name, tr.trigger\_id, tr.tname

from DG\_Base\_Trigger\_Table tr

inner join DG\_Tables t on t.Table\_ID = tr.Table\_ID

inner join DG\_Schema s on s.schema\_ID = t.Database\_ID

group by s.schema\_name, t.table\_name, tr.trigger\_id, tr.tname

go

exec SchemaTableTriggers

1. For every database/schema, list schema, dept, role, tables

create or alter procedure DeptTableInformation

as

select t.table\_name as TableName, s.schema\_name, r.role\_name as Roles, un.unit\_name as Dept

from DG\_tables t

inner join DG\_Unit\_Role\_Table ur on ur.Table\_id = t.Table\_ID

inner join DG\_Schema s on s.schema\_ID = t.Database\_ID

inner join DG\_Role r on r.Role\_ID = ur.Role\_ID

inner join DG\_Unit un on un.Unit\_ID = ur.Unit\_ID

group by t.Table\_Name, s.schema\_name, r.Role\_Name, un.Unit\_Name

go

exec DeptTableInformation

1. For every database/schema, list schemas equipped with triggers to manage change in dept/tables

create or alter procedure TriggersInDept as

select s.schema\_name, t.table\_name, tr.tname, u.unit\_name as Dept

from DG\_Unit u

inner join DG\_Unit\_Role\_Table un on un.Unit\_ID = u.Unit\_ID

inner join DG\_Base\_Trigger\_Table tr on tr.Table\_ID = un.Table\_id

inner join DG\_Tables t on t.Table\_ID = tr.Table\_ID

inner join DG\_Schema s on s.schema\_ID = t.Database\_ID

group by s.schema\_name, t.table\_name, tr.tname, u.unit\_name

go

exec TriggersInDept