COLLIE

USE ROLE SYSADMIN;

CREATE DATABASE "ACCELERATOR\_DB";

CREATE SCHEMA "ACCELERATOR\_DB"."RBAC";

GRANT USAGE ON DATABASE ACCELERATOR\_DB TO ROLE SECURITYADMIN;

GRANT USAGE,CREATE TEMPORARY TABLE ON SCHEMA RBAC TO ROLE SECURITYADMIN;

USE ROLE SECURITYADMIN;

USE SCHEMA ACCELERATOR\_DB.RBAC;

GRANT CREATE TAG ON SCHEMA ACCELERATOR\_DB.RBAC TO ROLE SECURITYADMIN;

CREATE TAG environment ALLOWED\_VALUES 'dev','qa', 'prod', 'system';

CREATE TAG role\_category ALLOWED\_VALUES 'fr','ar';

CREATE TAG role\_type ALLOWED\_VALUES 'ro','rw','all';

USE ROLE ACCOUNTADMIN;

ALTER ROLE ACCOUNTADMIN SET TAG environment = 'system';

ALTER ROLE SECURITYADMIN SET TAG environment = 'system';

ALTER ROLE USERADMIN SET TAG environment = 'system';

ALTER ROLE SYSADMIN SET TAG environment = 'system';

ALTER ROLE PUBLIC SET TAG environment = 'system';

ALTER ROLE ORGADMIN SET TAG environment = 'system';

USE ROLE SYSADMIN;

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_CREATE\_ROLE"("EXEC\_ROLE" VARCHAR(16777216), "ROLE\_NAME" VARCHAR(16777216), "COMMENTS" VARCHAR(16777216), "ENV" VARCHAR(16777216))

RETURNS VARIANT

LANGUAGE JAVASCRIPT

STRICT

COMMENT='Stored Procedure to create ROLE.'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

var inp\_role\_name = ROLE\_NAME.toLocaleUpperCase();

var inp\_comments=COMMENTS.toLocaleUpperCase();

var inp\_env=ENV.toLocaleLowerCase();

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

snowflake.execute({sqlText: `CREATE ROLE ${inp\_role\_name} COMMENT=''${inp\_comments}'' with tag(environment=''${inp\_env}'')`});

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''Role '' + inp\_role\_name + '' created successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_CREATE\_USER"("EXEC\_ROLE" VARCHAR(16777216), "DISPLAY\_NAME" VARCHAR(16777216), "LOGIN\_NAME" VARCHAR(16777216))

RETURNS VARIANT

LANGUAGE JAVASCRIPT

COMMENT='Stored Procedure to CREATE USER to build the RBAC hierarchy.'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

var display\_name = DISPLAY\_NAME.toLocaleUpperCase();

var login\_name= LOGIN\_NAME.toLocaleUpperCase();

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

snowflake.execute({sqlText: `CREATE USER ${display\_name} LOGIN\_NAME = ${login\_name} DISPLAY\_NAME = ${display\_name}`});

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''User '' + display\_name + '' created successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_CREATE\_WAREHOUSE"("EXEC\_ROLE" VARCHAR(16777216), "WAREHOUSE\_NAME" VARCHAR(16777216), "WAREHOUSE\_SIZE" VARCHAR(16777216), "MAX\_CLUSTER\_COUNT" FLOAT, "MIN\_CLUSTER\_COUNT" FLOAT, "SCALING\_POLICY" VARCHAR(16777216), "AUTO\_SUSPEND" FLOAT, "AUTO\_RESUME" VARCHAR(16777216), "INITIALLY\_SUSPENDED" VARCHAR(16777216))

RETURNS VARIANT

LANGUAGE JAVASCRIPT

COMMENT='Generic Script to Create warehouse'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

var wh\_name= WAREHOUSE\_NAME.toLocaleUpperCase();

var wh\_size= WAREHOUSE\_SIZE.toLocaleUpperCase();

var max\_cluster= MAX\_CLUSTER\_COUNT;

var min\_cluster=MIN\_CLUSTER\_COUNT;

var scaling\_policy=SCALING\_POLICY.toLocaleUpperCase();

var auto\_suspend=AUTO\_SUSPEND;

var auto\_resume=AUTO\_RESUME.toLocaleUpperCase();

var initailly\_suspend=INITIALLY\_SUSPENDED.toLocaleUpperCase();

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

if(auto\_suspend==0)

{

auto\_suspend=null;

}

snowflake.execute({sqlText: `CREATE WAREHOUSE ${wh\_name} WAREHOUSE\_SIZE = ${wh\_size} AUTO\_SUSPEND = ${auto\_suspend} AUTO\_RESUME = ${auto\_resume} MIN\_CLUSTER\_COUNT = ${min\_cluster} MAX\_CLUSTER\_COUNT = ${max\_cluster} SCALING\_POLICY = ${scaling\_policy}`});

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''WAREHOUSE '' + wh\_name + '' created successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_DROP\_ROLE"("EXEC\_ROLE" VARCHAR(16777216), "ROLE\_NAME" VARCHAR(16777216), "IS\_CONFIRMED" FLOAT)

RETURNS VARIANT

LANGUAGE JAVASCRIPT

COMMENT='Stored Procedure to drop the ROLE from snowflake instance.'

EXECUTE AS CALLER

AS '

var count=0;

var is\_confirmed=IS\_CONFIRMED;

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

var role\_name = ROLE\_NAME.toLocaleUpperCase();

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

if(is\_confirmed==0)

{

snowflake.execute({sqlText: `USE ROLE SYSADMIN`});

snowflake.execute({sqlText: `SHOW GRANTS TO ROLE IDENTIFIER(''${role\_name}'')`});

var rs = snowflake.execute({sqlText: `SELECT "name" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID())) WHERE "granted\_to" = ''ROLE'' AND "granted\_on" = ''ROLE'' AND "privilege"=''USAGE'';`});

while(rs.next())

{

count=count+1;

}

if(count==0)

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

snowflake.execute({sqlText: `DROP ROLE ${role\_name}`});

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''Role '' + role\_name + '' dropped successfully!''};

return table\_as\_json;

}

else{

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''WARNING: Role '' + role\_name + '' has dependencies, please revoke the dependencies before dropping the role!''};

return table\_as\_json;

}

}

else

{

snowflake.execute({sqlText: `DROP ROLE ${role\_name}`});

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''Role '' + role\_name + '' dropped successfully!''};

return table\_as\_json;

}

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_DROP\_USER"("EXEC\_ROLE" VARCHAR(16777216), "DISPLAY\_NAME" ARRAY)

RETURNS VARIANT

LANGUAGE JAVASCRIPT

COMMENT='Stored Procedure to drop the USER from snowflake instance.'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

for (var col\_num = 0; col\_num < DISPLAY\_NAME.length; col\_num = col\_num + 1)

{

var display\_name = DISPLAY\_NAME[col\_num].toLocaleUpperCase();

snowflake.execute({sqlText: `DROP USER ${display\_name}`});

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response":''User '' + DISPLAY\_NAME + '' dropped successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_DROP\_WAREHOUSE"("EXEC\_ROLE" VARCHAR(16777216), "WAREHOUSE\_NAME" ARRAY)

RETURNS VARIANT

LANGUAGE JAVASCRIPT

COMMENT='Stored Procedure to drop the WAREHOUSE from snowflake instance.'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

for (var col\_num = 0; col\_num < WAREHOUSE\_NAME.length; col\_num = col\_num + 1)

{

var wh\_name = WAREHOUSE\_NAME[col\_num].toLocaleUpperCase();

snowflake.execute({sqlText: `DROP WAREHOUSE ${wh\_name}`});

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''Warehouse '' + WAREHOUSE\_NAME + '' dropped successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_GRANT\_PRIVS\_TO\_ROLE"("EXEC\_ROLE" VARCHAR(16777216), "PRIVILEGES\_LIST" VARCHAR(16777216), "ON\_OBJECT\_LEVEL" VARCHAR(16777216), "ON\_OBJECT\_NAME" ARRAY, "IN\_OBJECT\_LEVEL" VARCHAR(16777216), "IN\_OBJECT\_NAME" VARCHAR(16777216), "TARGET\_ROLE\_NAME" ARRAY)

RETURNS VARIANT

LANGUAGE JAVASCRIPT

COMMENT='Stored Procedure to grant ROLE to a ROLE or a USER to build the RBAC hierarchy.'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

var inp\_privileges = PRIVILEGES\_LIST.toLocaleUpperCase();

if (!ON\_OBJECT\_LEVEL) { var inp\_on\_object\_level = '''' }

else {var inp\_on\_object\_level = ON\_OBJECT\_LEVEL.toLocaleUpperCase();}

if (!IN\_OBJECT\_LEVEL) { var inp\_in\_object\_level = '''' }

else {var inp\_in\_object\_level = IN\_OBJECT\_LEVEL.toLocaleUpperCase();}

if (!IN\_OBJECT\_NAME) { var inp\_in\_object\_name = '''' }

else {var inp\_in\_object\_name = IN\_OBJECT\_NAME.toLocaleUpperCase();}

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

for (var col\_num1 = 0; col\_num1 < TARGET\_ROLE\_NAME.length; col\_num1 = col\_num1 + 1)

{

var inp\_target\_role\_name = TARGET\_ROLE\_NAME[col\_num1].toLocaleUpperCase();

if (!inp\_in\_object\_level || !inp\_in\_object\_name)

{

for (var col\_num = 0; col\_num < ON\_OBJECT\_NAME.length; col\_num = col\_num + 1)

{

if (!ON\_OBJECT\_NAME[col\_num]) {var inp\_on\_object\_name = '''' }

else {var inp\_on\_object\_name = ON\_OBJECT\_NAME[col\_num].toLocaleUpperCase();}

var resp=snowflake.execute({sqlText: `GRANT ${inp\_privileges} ON ${inp\_on\_object\_level} ${inp\_on\_object\_name} TO ROLE ${inp\_target\_role\_name}`});

while(resp.next())

{

if(resp.getColumnValue(1)=="Grant not executed: Insufficient privileges.")

{

throw new Error("Role doesn''t have enough privileges to grant privileges");

}

else

{

continue;

}

}

continue;

}

}

else

{

for (var col\_num = 0; col\_num < ON\_OBJECT\_NAME.length; col\_num = col\_num + 1)

{

if (!ON\_OBJECT\_NAME[col\_num]) {var inp\_on\_object\_name = '''' }

else {var inp\_on\_object\_name = ON\_OBJECT\_NAME[col\_num].toLocaleUpperCase();}

var resp=snowflake.execute({sqlText: `GRANT ${inp\_privileges} ON ${inp\_on\_object\_level} ${inp\_on\_object\_name} IN ${inp\_in\_object\_level} ${inp\_in\_object\_name} TO ROLE ${inp\_target\_role\_name}`});

while (resp.next())

{

if(resp.getColumnValue(1)=="Grant not executed: Insufficient privileges.")

{

throw new Error("Role doesn''t have enough privileges to grant privileges");

}

else

{

continue;

}

}

continue;

}

}

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''Privileges are granted to the ROLE '' + TARGET\_ROLE\_NAME + '' successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_GRANT\_PRIVS\_TO\_ROLE\_TEST"("EXEC\_ROLE" VARCHAR(16777216), "PRIVILEGES\_LIST" VARCHAR(16777216), "ON\_OBJECT\_LEVEL" VARCHAR(16777216), "ON\_OBJECT\_NAME" ARRAY, "IN\_OBJECT\_LEVEL" VARCHAR(16777216), "IN\_OBJECT\_NAME" VARCHAR(16777216), "TARGET\_ROLE\_NAME" ARRAY)

RETURNS VARIANT

LANGUAGE JAVASCRIPT

COMMENT='Stored Procedure to grant ROLE to a ROLE or a USER to build the RBAC hierarchy.'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

var inp\_privileges = PRIVILEGES\_LIST.toLocaleUpperCase();

if (!ON\_OBJECT\_LEVEL) { var inp\_on\_object\_level = '''' }

else {var inp\_on\_object\_level = ON\_OBJECT\_LEVEL.toLocaleUpperCase();}

if (!IN\_OBJECT\_LEVEL) { var inp\_in\_object\_level = '''' }

else {var inp\_in\_object\_level = IN\_OBJECT\_LEVEL.toLocaleUpperCase();}

if (!IN\_OBJECT\_NAME) { var inp\_in\_object\_name = '''' }

else {var inp\_in\_object\_name = IN\_OBJECT\_NAME.toLocaleUpperCase();}

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

for (var col\_num1 = 0; col\_num1 < TARGET\_ROLE\_NAME.length; col\_num1 = col\_num1 + 1)

{

var inp\_target\_role\_name = TARGET\_ROLE\_NAME[col\_num1].toLocaleUpperCase();

if (!inp\_in\_object\_level || !inp\_in\_object\_name)

{

for (var col\_num = 0; col\_num < ON\_OBJECT\_NAME.length; col\_num = col\_num + 1)

{

if (!ON\_OBJECT\_NAME[col\_num]) {var inp\_on\_object\_name = '''' }

else {var inp\_on\_object\_name = ON\_OBJECT\_NAME[col\_num].toLocaleUpperCase();}

var resp=snowflake.execute({sqlText: `GRANT ${inp\_privileges} ON ${inp\_on\_object\_level} ${inp\_on\_object\_name} TO ROLE ${inp\_target\_role\_name}`});

while(resp.next())

{

if(resp.getColumnValue(1)=="Grant not executed: Insufficient privileges.")

{

throw new Error("Role doesn''t have enough privileges to grant privileges");

}

else

{

continue;

}

}

continue;

}

}

else

{

for (var col\_num = 0; col\_num < ON\_OBJECT\_NAME.length; col\_num = col\_num + 1)

{

if (!ON\_OBJECT\_NAME[col\_num]) {var inp\_on\_object\_name = '''' }

else {var inp\_on\_object\_name = ON\_OBJECT\_NAME[col\_num].toLocaleUpperCase();}

var resp=snowflake.execute({sqlText: `GRANT ${inp\_privileges} ON ${inp\_on\_object\_level} ${inp\_on\_object\_name} IN ${inp\_in\_object\_level} ${inp\_in\_object\_name} TO ROLE ${inp\_target\_role\_name}`});

while (resp.next())

{

if(resp.getColumnValue(1)=="Grant not executed: Insufficient privileges.")

{

throw new Error("Role doesn''t have enough privileges to grant privileges");

}

else

{

continue;

}

}

continue;

}

}

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''Privileges are granted to the ROLE '' + TARGET\_ROLE\_NAME + '' successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_GRANT\_ROLE\_TO\_ROLE"("EXEC\_ROLE" VARCHAR(16777216), "ROLE\_NAME" VARCHAR(16777216), "TARGET\_NAME" VARCHAR(16777216))

RETURNS VARIANT

LANGUAGE JAVASCRIPT

STRICT

COMMENT='Stored Procedure to grant ROLE to a ROLE or a USER to build the RBAC hierarchy.'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

var inp\_role\_name = ROLE\_NAME.toLocaleUpperCase();

var inp\_target\_name = TARGET\_NAME.toLocaleUpperCase();

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

var resp=snowflake.execute({sqlText: `GRANT ROLE ${inp\_role\_name} TO ROLE ${inp\_target\_name}`});

while (resp.next())

{

if(resp.getColumnValue(1)=="Grant not executed: Insufficient privileges.")

{

throw new Error("Role doesn''t have enough privileges to grant role");

}

else

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''ROLE '' + inp\_role\_name + '' is assigned to Role '' + '' '' + inp\_target\_name +'' successfully!''};

return table\_as\_json;

}

}

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_GRANT\_ROLE\_TO\_USER"("EXEC\_ROLE" VARCHAR(16777216), "ROLE\_NAME" ARRAY, "TARGET\_NAME" ARRAY)

RETURNS VARIANT

LANGUAGE JAVASCRIPT

STRICT

COMMENT='Stored Procedure to grant ROLE to a ROLE or a USER to build the RBAC hierarchy.'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

for (var col\_num = 0; col\_num < ROLE\_NAME.length; col\_num = col\_num + 1)

{

var inp\_role\_name = ROLE\_NAME[col\_num].toLocaleUpperCase();

for (var col\_num1 = 0; col\_num1 < TARGET\_NAME.length; col\_num1 = col\_num1 + 1)

{

var inp\_target\_name = TARGET\_NAME[col\_num1].toLocaleUpperCase();

var resp=snowflake.execute({sqlText: `GRANT ROLE ${inp\_role\_name} TO USER ${inp\_target\_name}`});

while (resp.next())

{

if(resp.getColumnValue(1)=="Grant not executed: Insufficient privileges.")

{

throw new Error("Role doesn''t have enough privileges to grant role");

}

}

}

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''ROLE '' + ROLE\_NAME + '' is assigned to User ''+ '' '' + TARGET\_NAME +'' successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

-- Deprecated due to slow performance:

-- CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_INDEPENDENT\_ROLES"()

-- RETURNS VARIANT

-- LANGUAGE JAVASCRIPT

-- EXECUTE AS CALLER

-- AS '

-- var test=[];

-- var test1=[];

-- var array\_of\_rows = [];

-- var table\_as\_json = {};

-- snowflake.execute({sqlText: `CREATE OR REPLACE TEMPORARY TABLE TBL (CHILD\_ROLE VARCHAR,PARENT\_ROLE VARCHAR,ENVIRONMENT VARCHAR);`});

-- try

-- {

-- var rs = snowflake.execute({sqlText: `show roles;`});

-- var insert\_records\_array = [];

-- var temp=snowflake.execute({sqlText: `SELECT "name" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID())) WHERE

-- ("name" <> ''ACCOUNTADMIN'' and "granted\_to\_roles"=0 and "granted\_roles"=0) or

-- ("name" <> ''ACCOUNTADMIN'' and "granted\_to\_roles"=0 and "granted\_roles">0);`});

-- while (temp.next())

-- {

-- var in\_role\_name = temp.getColumnValue(1);

-- insert\_records\_array.push(`(''${in\_role\_name}'',''NULL'',''NULL'')`);

-- test.push(in\_role\_name);

-- test1.push(in\_role\_name);

-- }

-- for (var col\_num = 0; col\_num < test.length; col\_num = col\_num + 1)

-- {

-- var in\_role\_name0=test[col\_num];

-- snowflake.execute({sqlText: `SHOW GRANTS TO ROLE IDENTIFIER(''${in\_role\_name0}'')`});

-- var rs1 = snowflake.execute({sqlText: `SELECT "name" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID())) WHERE "granted\_to" = ''ROLE'' AND "granted\_on" = ''ROLE'' AND "privilege"=''USAGE'';`});

-- while (rs1.next()){

-- var var1=rs1.getColumnValue(1);

-- if(var1){

-- test.push(var1);

-- insert\_records\_array.push(`(''${var1}'',''${in\_role\_name0}'',''NULL'')`);

-- }

-- }

-- }

-- if (insert\_records\_array.length != 0) {

-- snowflake.execute({ sqlText: `INSERT INTO TBL VALUES` + insert\_records\_array.join() + `;` });

-- }

-- for (var col\_num = 0; col\_num < test.length; col\_num = col\_num + 1)

-- {

-- var in\_role\_name = test[col\_num];

-- var temp=snowflake.execute({sqlText: `SELECT tag\_value FROM TABLE(information\_schema.tag\_references(''${in\_role\_name}'', ''ROLE'')) where tag\_name=''ENVIRONMENT'';`});

-- if(temp.next()){var env=temp.getColumnValue(1);}else{var env='''';}

-- snowflake.execute({sqlText: `UPDATE TBL SET ENVIRONMENT = ''${env}'' WHERE CHILD\_ROLE=''${in\_role\_name}''`});

-- }

-- var row\_num = 1;

-- for (var col\_num = 0; col\_num < test1.length; col\_num = col\_num + 1)

-- {

-- var in\_role\_name0=test1[col\_num];

-- var rs2=snowflake.execute({sqlText: `SELECT

-- distinct

-- LEVEL-1 as level,

-- CHILD\_ROLE,

-- PARENT\_ROLE,

-- ENVIRONMENT

-- FROM TBL

-- START WITH CHILD\_ROLE = ''${in\_role\_name0}''

-- CONNECT BY PARENT\_ROLE = PRIOR CHILD\_ROLE

-- ORDER BY LEVEL`});

-- while (rs2.next())

-- { var row\_as\_json = {};

-- row\_as\_json[''level''] = rs2.getColumnValue(1);

-- row\_as\_json[''childRole''] = rs2.getColumnValue(2);

-- row\_as\_json[''parentRole''] = rs2.getColumnValue(3);

-- row\_as\_json[''environment''] = rs2.getColumnValue(4);

-- array\_of\_rows.push(row\_as\_json);

-- ++row\_num;

-- }

-- }

-- var table\_as\_json = {};

-- table\_as\_json = { "flag" : 1,"response":array\_of\_rows};

-- return table\_as\_json;

-- }

-- catch(err)

-- {

-- var table\_as\_json = {};

-- table\_as\_json = { "flag" : 0,"response": err.message};

-- return table\_as\_json;

-- }

-- ';

--v2.0

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC.SP\_INDEPENDENT\_ROLES()

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var array\_of\_rows = [];

snowflake.execute({ sqlText: `CREATE OR REPLACE TEMPORARY TABLE TBL (CHILD\_ROLE VARCHAR,PARENT\_ROLE VARCHAR,ENVIRONMENT VARCHAR);` });

try {

snowflake.execute({ sqlText: `show roles;` });

var rs = snowflake.execute({ sqlText: `SELECT "name", "granted\_to\_roles" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID())) ORDER BY "created\_on" ASC;` });

var insert\_records\_array = [];

var disconnected\_roles\_array = [];

while (rs.next()) {

var role\_name = rs.getColumnValue(1);

var temp\_tag=snowflake.execute({sqlText: `SELECT tag\_value FROM TABLE(information\_schema.tag\_references(''${role\_name}'', ''ROLE'')) where tag\_name=''ENVIRONMENT'';`});

var env='''';

if(temp\_tag.next()){

env= temp\_tag.getColumnValue(1);

}

if (rs.getColumnValue(2) > 0) {

snowflake.execute({ sqlText: `SHOW GRANTS OF ROLE IDENTIFIER(''${role\_name}'')` });

var rs1 = snowflake.execute({ sqlText: `SELECT "grantee\_name" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID())) WHERE "granted\_to"= ''ROLE'' ORDER BY "created\_on" ASC;` });

while (rs1.next()) {

var parent\_role\_name=rs1.getColumnValue(1);

insert\_records\_array.push(`(''${ role\_name}'',''${parent\_role\_name}'',''${env}'')`);

}

}

else {

insert\_records\_array.push(`(''${role\_name}'',''NULL'',''${env}'')`);

if (role\_name != ''ACCOUNTADMIN'') {

disconnected\_roles\_array.push(role\_name);

}

}

}

if (insert\_records\_array.length != 0) {

snowflake.execute({ sqlText: `INSERT INTO TBL VALUES` + insert\_records\_array.join() + `;` });

}

for (var col\_num = 0; col\_num < disconnected\_roles\_array.length; col\_num = col\_num + 1)

{

var in\_role\_name0=disconnected\_roles\_array[col\_num];

var rs2=snowflake.execute({sqlText: `SELECT

distinct

LEVEL-1 as level,

CHILD\_ROLE,

PARENT\_ROLE,

ENVIRONMENT

FROM TBL

START WITH CHILD\_ROLE = ''${in\_role\_name0}''

CONNECT BY PARENT\_ROLE = PRIOR CHILD\_ROLE

ORDER BY LEVEL`});

var row\_num = 1;

while (rs2.next())

{

var row\_as\_json = {};

row\_as\_json[''level''] = rs2.getColumnValue(1);

row\_as\_json[''childRole''] = rs2.getColumnValue(2);

row\_as\_json[''parentRole''] = rs2.getColumnValue(3);

row\_as\_json[''environment''] = rs2.getColumnValue(4);

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

}

var table\_as\_json = {};

table\_as\_json = { "flag": 1, "response": array\_of\_rows };

return table\_as\_json;

}

catch (err) {

var table\_as\_json = {};

table\_as\_json = { "flag": 0, "response": err.message };

return table\_as\_json;

}

';

-- Deprecated due to slow performance:

-- CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RBAC\_HIERARCHY"()

-- RETURNS VARIANT

-- LANGUAGE JAVASCRIPT

-- EXECUTE AS CALLER

-- AS '

-- var array\_of\_rows = [];

-- snowflake.execute({ sqlText: `CREATE OR REPLACE TEMPORARY TABLE TBL (CHILD\_ROLE VARCHAR,PARENT\_ROLE VARCHAR,ENVIRONMENT VARCHAR);` });

-- snowflake.execute({ sqlText: `INSERT INTO TBL VALUES(''ACCOUNTADMIN'',''NULL'',''system'');` });

-- try {

-- var rs = snowflake.execute({ sqlText: `show roles;` });

-- var insert\_records\_array = [];

-- while (rs.next()) {

-- var in\_role\_name = rs.getColumnValue(2);

-- snowflake.execute({ sqlText: `SHOW GRANTS TO ROLE IDENTIFIER(''${in\_role\_name}'')` });

-- var rs1 = snowflake.execute({ sqlText: `SELECT "name" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID())) WHERE "granted\_to" = ''ROLE'' AND "granted\_on" = ''ROLE'' AND "privilege"=''USAGE'';` });

-- while (rs1.next()) {

-- var var1 = rs1.getColumnValue(1);

-- var var1=rs1.getColumnValue(1);

-- var temp=snowflake.execute({sqlText: `SELECT tag\_value FROM TABLE(information\_schema.tag\_references(''${var1}'', ''ROLE'')) where tag\_name=''ENVIRONMENT'';`});

-- if(temp.next()){var env=temp.getColumnValue(1);}else{var env=''''}

-- insert\_records\_array.push(`(''${var1}'',''${in\_role\_name}'',''${env}'')`);

-- }

-- }

-- if (insert\_records\_array.length != 0) {

-- snowflake.execute({ sqlText: `INSERT INTO TBL VALUES` + insert\_records\_array.join() + `;` });

-- }

-- var rs1 = snowflake.execute({

-- sqlText: `

-- SELECT

-- LEVEL-1 as level,

-- CHILD\_ROLE,

-- PARENT\_ROLE,

-- ENVIRONMENT

-- FROM TBL

-- START WITH CHILD\_ROLE = ''ACCOUNTADMIN''

-- CONNECT BY PARENT\_ROLE = PRIOR CHILD\_ROLE

-- ORDER BY LEVEL`});

-- var row\_num = 1;

-- while (rs1.next()) {

-- var row\_as\_json = {};

-- for (var col\_num = 0; col\_num < 3; col\_num = col\_num + 1) {

-- row\_as\_json[''LEVEL''] = rs1.getColumnValue(1);

-- row\_as\_json[''CHILD\_ROLE''] = rs1.getColumnValue(2);

-- row\_as\_json[''PARENT\_ROLE''] = rs1.getColumnValue(3);

-- row\_as\_json[''ENVIRONMENT''] = rs1.getColumnValue(4);

-- }

-- array\_of\_rows.push(row\_as\_json);

-- ++row\_num;

-- }

-- var table\_as\_json = {};

-- table\_as\_json = { "flag": 1, "response": array\_of\_rows };

-- return table\_as\_json;

-- }

-- catch (err) {

-- var table\_as\_json = {};

-- table\_as\_json = { "flag": 0, "response": err.message };

-- return table\_as\_json;

-- }

-- ';

--v2.0

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC.SP\_RBAC\_HIERARCHY()

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var array\_of\_rows = [];

snowflake.execute({ sqlText: `CREATE OR REPLACE TEMPORARY TABLE TBL (CHILD\_ROLE VARCHAR,PARENT\_ROLE VARCHAR,ENVIRONMENT VARCHAR);` });

try {

snowflake.execute({ sqlText: `show roles;` });

var rs = snowflake.execute({ sqlText: `SELECT "name", "granted\_to\_roles" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID())) ORDER BY "created\_on" ASC;` });

var insert\_records\_array = [];

while (rs.next()) {

var role\_name = rs.getColumnValue(1);

var temp\_tag=snowflake.execute({sqlText: `SELECT tag\_value FROM TABLE(information\_schema.tag\_references(''${role\_name}'', ''ROLE'')) where tag\_name=''ENVIRONMENT'';`});

var env='''';

if(temp\_tag.next()){

env= temp\_tag.getColumnValue(1);

}

if (rs.getColumnValue(2) > 0) {

snowflake.execute({ sqlText: `SHOW GRANTS OF ROLE IDENTIFIER(''${role\_name}'')` });

var rs1 = snowflake.execute({ sqlText: `SELECT "grantee\_name" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID())) WHERE "granted\_to"= ''ROLE'' ORDER BY "created\_on" ASC;` });

while (rs1.next()) {

var parent\_role\_name=rs1.getColumnValue(1);

insert\_records\_array.push(`(''${ role\_name}'',''${parent\_role\_name}'',''${env}'')`);

}

}

else {

insert\_records\_array.push(`(''${role\_name}'',''NULL'',''${env}'')`);

}

}

if (insert\_records\_array.length != 0) {

snowflake.execute({ sqlText: `INSERT INTO TBL VALUES` + insert\_records\_array.join() + `;` });

}

var rs1 = snowflake.execute({

sqlText: `

SELECT distinct

LEVEL-1 as level,

CHILD\_ROLE,

PARENT\_ROLE,

ENVIRONMENT

FROM TBL

START WITH CHILD\_ROLE = ''ACCOUNTADMIN''

CONNECT BY PARENT\_ROLE = PRIOR CHILD\_ROLE

ORDER BY LEVEL`});

var row\_num = 1;

while (rs1.next()) {

var row\_as\_json = {};

row\_as\_json[''LEVEL''] = rs1.getColumnValue(1);

row\_as\_json[''CHILD\_ROLE''] = rs1.getColumnValue(2);

row\_as\_json[''PARENT\_ROLE''] = rs1.getColumnValue(3);

row\_as\_json[''ENVIRONMENT''] = rs1.getColumnValue(4);

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

var table\_as\_json = {};

table\_as\_json = { "flag": 1, "response": array\_of\_rows };

return table\_as\_json;

}

catch (err) {

var table\_as\_json = {};

table\_as\_json = { "flag": 0, "response": err.message };

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_REPLICATE\_PRIV\_TO\_USER"("EXEC\_ROLE" VARCHAR(16777216), "TARGET\_USER" ARRAY, "MODEL\_USER" VARCHAR(16777216))

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var exec\_role = EXEC\_ROLE.toLocaleUpperCase();

var model\_user = MODEL\_USER.toLocaleUpperCase();

try

{

snowflake.execute({sqlText: `USE ROLE ${exec\_role}`});

try{

snowflake.execute({sqlText: `USE ROLE SYSADMIN`});

snowflake.execute({sqlText: `SHOW GRANTS TO USER ${model\_user}`});

var rs = snowflake.execute({sqlText: `SELECT "role" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID())) WHERE "granted\_to" = ''USER'';`});

snowflake.execute({sqlText: `USE ROLE ${exec\_role}`});

while (rs.next()) {

var var1=rs.getColumnValue(1);

for (var col\_num = 0; col\_num < TARGET\_USER.length; col\_num = col\_num + 1)

{

var target = TARGET\_USER[col\_num].toLocaleUpperCase();

snowflake.execute({sqlText: `GRANT ROLE ${var1} TO USER ${target};`});

}

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''Privileges of '' + model\_user + '' is replicated to User ''+ '' '' + TARGET\_USER +'' successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RETURN\_DATABASES"()

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var row\_as\_json = {};

var array\_of\_rows = [];

var COL\_NAMES=[''NAME'',''OWNER''];

try{

snowflake.execute({sqlText: `SHOW DATABASES`});

var rs = snowflake.execute({sqlText: `SELECT "name","owner" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID()))`});

var row\_num = 1;

while (rs.next()) {

row\_as\_json = {};

for (var col\_num = 0; col\_num < COL\_NAMES.length; col\_num = col\_num + 1) {

var col\_name = COL\_NAMES[col\_num];

row\_as\_json[col\_name] = rs.getColumnValue(col\_num + 1);

}

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response" : array\_of\_rows};

return table\_as\_json;

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RETURN\_INTEGRATIONS"()

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var row\_as\_json = {};

var array\_of\_rows = [];

var COL\_NAMES=[''NAME'',''TYPE''];

try{

snowflake.execute({sqlText: `SHOW INTEGRATIONS`});

var rs = snowflake.execute({sqlText: `SELECT "name","type" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID()))`});

var row\_num = 1;

while (rs.next()) {

row\_as\_json = {};

for (var col\_num = 0; col\_num < COL\_NAMES.length; col\_num = col\_num + 1) {

var col\_name = COL\_NAMES[col\_num];

row\_as\_json[col\_name] = rs.getColumnValue(col\_num + 1);

}

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response" : array\_of\_rows};

return table\_as\_json;

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RETURN\_OBJECT\_LIST"("DB\_NAME" VARCHAR(16777216), "SCHEMA\_NAME" VARCHAR(16777216), "OBJECT\_TYPE" VARCHAR(16777216))

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var inp\_db\_name = DB\_NAME.toLocaleUpperCase();

var inp\_schema\_name = SCHEMA\_NAME.toLocaleUpperCase();

var inp\_object\_type = OBJECT\_TYPE.toLocaleUpperCase();

var row\_as\_json = {};

var array\_of\_rows = [];

try{

if (inp\_object\_type=="STAGES")

{

snowflake.execute({sqlText: `SHOW ${inp\_object\_type} in schema ${inp\_db\_name}.${inp\_schema\_name}`});

var rs = snowflake.execute({sqlText: `SELECT "name","type" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID()))`});

var row\_num = 1;

while (rs.next()) {

row\_as\_json = {};

row\_as\_json[''name''] = rs.getColumnValue(1);

row\_as\_json[''type''] = rs.getColumnValue(2);

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

}

else{

snowflake.execute({sqlText: `SHOW ${inp\_object\_type} in schema ${inp\_db\_name}.${inp\_schema\_name}`});

var rs = snowflake.execute({sqlText: `SELECT "name" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID()))`});

var row\_num = 1;

while (rs.next()) {

row\_as\_json = {};

row\_as\_json[''name''] = rs.getColumnValue(1);

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response" : array\_of\_rows};

return table\_as\_json;

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RETURN\_RESOURCE\_MONITORS"()

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var row\_as\_json = {};

var array\_of\_rows = [];

var COL\_NAMES=[''NAME'',''CREDIT\_QUOTA''];

try{

snowflake.execute({sqlText: `SHOW RESOURCE MONITORS`});

var rs = snowflake.execute({sqlText: `SELECT "name","credit\_quota" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID()))`});

var row\_num = 1;

while (rs.next()) {

row\_as\_json = {};

for (var col\_num = 0; col\_num < COL\_NAMES.length; col\_num = col\_num + 1) {

var col\_name = COL\_NAMES[col\_num];

row\_as\_json[col\_name] = rs.getColumnValue(col\_num + 1);

}

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response" : array\_of\_rows};

return table\_as\_json;

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RETURN\_ROLES"()

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var row\_as\_json = {};

var array\_of\_rows = [];

var COL\_NAMES=[''NAME'',''OWNER''];

try{

snowflake.execute({sqlText: `SHOW ROLES`});

var rs = snowflake.execute({sqlText: `SELECT "name", "owner" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID()))`});

var row\_num = 1;

while (rs.next()) {

row\_as\_json = {};

for (var col\_num = 0; col\_num < COL\_NAMES.length; col\_num = col\_num + 1) {

var col\_name = COL\_NAMES[col\_num];

row\_as\_json[col\_name] = rs.getColumnValue(col\_num + 1);

}

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": array\_of\_rows};

return table\_as\_json;

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RETURN\_SCHEMAS"("DB\_NAME" VARCHAR(16777216))

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var inp\_database = DB\_NAME.toLocaleUpperCase();

var row\_as\_json = {};

var array\_of\_rows = [];

var COL\_NAMES=[''NAME'',''OWNER''];

try{

snowflake.execute({sqlText: `SHOW schemas IN DATABASE ${inp\_database};`});

var rs =snowflake.execute({sqlText: `SELECT "name","owner" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID()));`});

var row\_num = 1;

while (rs.next()) {

row\_as\_json = {};

for (var col\_num = 0; col\_num < COL\_NAMES.length; col\_num = col\_num + 1) {

var col\_name = COL\_NAMES[col\_num];

row\_as\_json[col\_name] = rs.getColumnValue(col\_num + 1);

}

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response" : array\_of\_rows};

return table\_as\_json;

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RETURN\_TAG\_VALUES"()

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var rownum=0;

var array\_of\_rows = [];

try

{

snowflake.execute({sqlText: `SHOW TAGS;`});

var tags=snowflake.execute({sqlText: `SELECT "name","allowed\_values" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID()));`});

while(tags.next())

{

var row\_as\_json = {};

row\_as\_json[''tag'']=tags.getColumnValue(1);

row\_as\_json[''tagValues'']=tags.getColumnValue(2);

array\_of\_rows.push(row\_as\_json);

++rownum;

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response":array\_of\_rows};

return table\_as\_json;

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RETURN\_USERS"()

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var row\_as\_json = {};

var array\_of\_rows = [];

var COL\_NAMES=[''NAME'',''LOGIN\_NAME''];

// snowflake.execute({sqlText: `USE ROLE SECURITYADMIN`});

try

{

snowflake.execute({sqlText: `SHOW USERS`});

var rs = snowflake.execute({sqlText: `SELECT "name","login\_name" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID()))`});

// Read each row and add it to the array we will return.

var row\_num = 1;

while (rs.next()) {

// Put each row in a variable of type JSON.

row\_as\_json = {};

// For each column in the row...

for (var col\_num = 0; col\_num < COL\_NAMES.length; col\_num = col\_num + 1) {

var col\_name = COL\_NAMES[col\_num];

row\_as\_json[col\_name] = rs.getColumnValue(col\_num + 1);

}

// Add the row to the array of rows.

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

// Put the array in a JSON variable (so it looks like a VARIANT to Snowflake). The key is "key1", and the value is the array that has the rows we want.

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response":array\_of\_rows};

return table\_as\_json;

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RETURN\_USERS\_OF\_ROLE"("ROLE\_NAME" VARCHAR(16777216))

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var inp\_role\_name = ROLE\_NAME.toLocaleUpperCase();

var array\_of\_rows = [];

try{

var usr=snowflake.execute({sqlText: `SHOW USERS`});

while (usr.next()) {

var usr1=usr.getColumnValue(1)

snowflake.execute({sqlText: `SHOW GRANTS TO USER ${usr1}`});

var rs = snowflake.execute({sqlText: `SELECT "grantee\_name" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID())) WHERE "granted\_to" = ''USER'' and "role"=''${inp\_role\_name}'';`});

while (rs.next()) {

array\_of\_rows.push(rs.getColumnValue(1));

}

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response":array\_of\_rows};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_RETURN\_WAREHOUSES"()

RETURNS VARIANT

LANGUAGE JAVASCRIPT

EXECUTE AS CALLER

AS '

var row\_as\_json = {};

var array\_of\_rows = [];

var COL\_NAMES=[''NAME'',''STATE'',''SIZE''];

try{

snowflake.execute({sqlText: `SHOW WAREHOUSES`});

var rs = snowflake.execute({sqlText: `SELECT "name","state","size" FROM TABLE(RESULT\_SCAN(LAST\_QUERY\_ID()))`});

var row\_num = 1;

while (rs.next()) {

row\_as\_json = {};

for (var col\_num = 0; col\_num < COL\_NAMES.length; col\_num = col\_num + 1) {

var col\_name = COL\_NAMES[col\_num];

row\_as\_json[col\_name] = rs.getColumnValue(col\_num + 1);

}

array\_of\_rows.push(row\_as\_json);

++row\_num;

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response":array\_of\_rows};

return table\_as\_json;

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_REVOKE\_PRIVS\_FROM\_ROLE"("EXEC\_ROLE" VARCHAR(16777216), "PRIVILEGES\_LIST" VARCHAR(16777216), "ON\_OBJECT\_LEVEL" VARCHAR(16777216), "ON\_OBJECT\_NAME" ARRAY, "IN\_OBJECT\_LEVEL" VARCHAR(16777216), "IN\_OBJECT\_NAME" VARCHAR(16777216), "TARGET\_ROLE\_NAME" ARRAY)

RETURNS VARIANT

LANGUAGE JAVASCRIPT

COMMENT='Stored Procedure to grant ROLE to a ROLE or a USER to build the RBAC hierarchy.'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

var inp\_privileges = PRIVILEGES\_LIST.toLocaleUpperCase();

if (!ON\_OBJECT\_LEVEL) {var inp\_on\_object\_level = '''' }

else {var inp\_on\_object\_level = ON\_OBJECT\_LEVEL.toLocaleUpperCase();}

if (!IN\_OBJECT\_LEVEL) {var inp\_in\_object\_level = '''' }

else {var inp\_in\_object\_level = IN\_OBJECT\_LEVEL.toLocaleUpperCase();}

if (!IN\_OBJECT\_NAME) {var inp\_in\_object\_name = '''' }

else {var inp\_in\_object\_name = IN\_OBJECT\_NAME.toLocaleUpperCase();}

try

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

for (var col\_num1 = 0; col\_num1 < TARGET\_ROLE\_NAME.length; col\_num1 = col\_num1 + 1)

{

var inp\_target\_role\_name = TARGET\_ROLE\_NAME[col\_num1].toLocaleUpperCase();

if (!inp\_in\_object\_level || !inp\_in\_object\_name)

{

for (var col\_num = 0; col\_num < ON\_OBJECT\_NAME.length; col\_num = col\_num + 1)

{

if (!ON\_OBJECT\_NAME[col\_num]) {var inp\_on\_object\_name = '''' }

else {var inp\_on\_object\_name = ON\_OBJECT\_NAME[col\_num].toLocaleUpperCase();}

var resp=snowflake.execute({sqlText: `REVOKE ${inp\_privileges} ON ${inp\_on\_object\_level} ${inp\_on\_object\_name} FROM ROLE ${inp\_target\_role\_name}`});

while (resp.next())

{

if(resp.getColumnValue(1)=="Revoke not executed: Insufficient privileges.")

{

throw new Error("Role doesn''t have enough privileges to revoke privileges");

}

else

{

continue;

}

}

continue;

}

}

else

{

for (var col\_num = 0; col\_num < ON\_OBJECT\_NAME.length; col\_num = col\_num + 1)

{

if (!ON\_OBJECT\_NAME[col\_num]) {var inp\_on\_object\_name = '''' }

else {var inp\_on\_object\_name = ON\_OBJECT\_NAME[col\_num].toLocaleUpperCase();}

var resp=snowflake.execute({sqlText: `REVOKE ${inp\_privileges} ON ${inp\_on\_object\_level} ${inp\_on\_object\_name} IN ${inp\_in\_object\_level} ${inp\_in\_object\_name} FROM ROLE ${inp\_target\_role\_name}`});

while (resp.next())

{

if(resp.getColumnValue(1)=="Revoke not executed: Insufficient privileges.")

{

throw new Error("Role doesn''t have enough privileges to revoke privileges");

}

else

{

continue;

}

}

continue;

}

}

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''Privileges are revoked from the ROLE '' + TARGET\_ROLE\_NAME + '' successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

CREATE OR REPLACE PROCEDURE ACCELERATOR\_DB.RBAC."SP\_REVOKE\_ROLE"("EXEC\_ROLE" VARCHAR(16777216), "ROLE\_NAME" ARRAY, "TARGET\_TYPE" VARCHAR(16777216), "TARGET\_NAME" ARRAY)

RETURNS VARIANT

LANGUAGE JAVASCRIPT

STRICT

COMMENT='Stored Procedure to revoke ROLE from a ROLE or a USER.'

EXECUTE AS CALLER

AS '

var inp\_exec\_role = EXEC\_ROLE.toLocaleUpperCase();

var inp\_target\_type = TARGET\_TYPE.toLocaleUpperCase();

try

{

if (inp\_target\_type == ''USER'' || inp\_target\_type == ''ROLE'')

{

snowflake.execute({sqlText: `USE ROLE ${inp\_exec\_role}`});

try

{

for (var col\_num = 0; col\_num < ROLE\_NAME.length; col\_num = col\_num + 1)

{

var inp\_role\_name = ROLE\_NAME[col\_num].toLocaleUpperCase();

for (var col\_num1 = 0; col\_num1 < TARGET\_NAME.length; col\_num1 = col\_num1 + 1)

{

var inp\_target\_name = TARGET\_NAME[col\_num1].toLocaleUpperCase();

var resp=snowflake.execute({sqlText: `REVOKE ROLE ${inp\_role\_name} FROM ${inp\_target\_type} ${inp\_target\_name}`});

while (resp.next())

{

if(resp.getColumnValue(1)=="Revoke not executed: Insufficient privileges.")

{

throw new Error("Role doesn''t have enough privileges to revoke role");

}

}

}

}

var table\_as\_json = {};

table\_as\_json = { "flag" : 1,"response": ''ROLE '' + ROLE\_NAME + '' is revoked from ''+ inp\_target\_type + '' '' + TARGET\_NAME +'' successfully!''};

return table\_as\_json;

}

catch(e)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": e.message};

return table\_as\_json;

}

}

else

{

throw new Error("Please enter a valid TARGET\_TYPE (USER/ROLE)");

}

}

catch(err)

{

var table\_as\_json = {};

table\_as\_json = { "flag" : 0,"response": err.message};

return table\_as\_json;

}

';

Accelerators : [Kipi Accelerators Master sheet](https://docs.google.com/spreadsheets/d/1cwuAZkhL5_-fRxlqr57BeRUZW2ektviWJ-vOBlGzPHc/preview)

Collie: [Collie Onboarding Resources](https://docs.google.com/document/d/1OAoQ7NfKH2ls35ygsj--ZYKN2eybzrEZ0uSRUkrDjhs/edit)