Capgemini

SAP HANA

Lesson Name: Native SQL

Lesson Objectives

After completing this lesson, participants will be able to -

- Know about native SQL using SAP HANA
  - Viewing Tables
  - Select and From
  - Where Clauses
  - Functions
  - Group By
  - Order By
  - Having
  - Top
  - Create
  - Insert
  - Update
  - Delete
  - Joins

Lesson Objectives

- Sub Selects
- Unions
- Drop
- Views
- Schemas
- Table Types
- Difference between classical open SQL and Native SQL

Native SQL using SAP HANA

VIEWING TABLES

- Syntax

  Select column1, column2 from (select column1,column2 from “table_name”)

- example of code is:

```
SELECT TOP 200 “CUSTOMERID” ,“CITYID”, “COUNTRYID”, “REGIONID”,
“COMPANYNAME”, “POSTALCODE”, “CITYNAME”, “COUNTRYNAME”,
“REGIONNAME” FROM ( SELECT * FROM “STS” . “DIMCUSTOMER” ) TMP
```

Native SQL using SAP HANA

SELECT AND FROM

- Syntax
  Select COLUMN1, COLUMN2 FROM “TABLE_NAME”

- example of code is:

```
SELECT * FROM “STSFLAT”.“STSCUSTOMERFLATFILE”

SELECT  COMPANYNAME  FROM  STS.DIMCUSTOMER
```

Native SQL using SAP HANA

WHERE

- Syntax

  SELECT COLUMN1 COLUMN 2 FROM “TABLE1”
  where CUSTOMID =‘2’

- example of code is:

SELECT * FROM “STS”.“DIMCUSTOMER” where CITYNAME = ‘LONDON’

SELECT * FROM “STS”.“DIMCUSTOMER” where CITYNAME like ‘R%’

Native SQL using SAP HANA

FUNCTIONS

- Syntax

  SELECT COUNT(10) FROM “TABLE1’ where CUSTOMID=‘2’

- example of code is:

SELECT COUNT(*) FROM “STS”.“DIMCUSTOMER” where COUNTRYNAME = ‘USA’

SELECT COUNT(QUANTITY) FROM “STS”.“DIMCUSTOMER” where COUNTRYNAME = ‘USA’

Native SQL using SAP HANA

GROUP BY

- Syntax

  select column1 as field_name from table_name GROUP BY column1

- example of code is:

```
SELECT  countryname as COUNTRY,
SUM(netsales) as TOTAL_SALES
FROM “STSFLAT”.“STSCUSTOMERFLATFILE”
GROUP BY countryname
HAVING sum(netsales) > 4000000
order by COUNTRYNAME
```

Native SQL using SAP HANA

ORDER BY

- Syntax
  SELECT COLUMN1 COLUM2 FROM “TABLE1” order by VARIABLE

- example of code is:

```
SELECT  * FROM “STS”.“DIMCUSTOMER” order by CUSTOMERID
```

Native SQL using SAP HANA

HAVING

- Syntax

  select column1 as field_name from table_name HAVING condition

- example of code is:

```
SELECT  countryname as COUNTRY,
SUM(netsales) as TOTAL_SALES
FROM "STSFLAT"."STSCUSTOMERFLATFILE"
GROUP BY countryname
HAVING sum(netsales) > 4000000
order by COUNTRYNAME
```

Native SQL using SAP HANA

TOP

- Syntax
  select TOP no_var column1 as field_name from table_name

- example of code is:

```
SELECT  TOP 5 countryname as COUNTRY,
SUM(netsales) as TOTAL_SALES
FROM "STSFLAT"."STSCUSTOMERFLATFILE"
GROUP BY countryname
HAVING sum(netsales) > 4000000
order by COUNTRYNAME
```

Native SQL using SAP HANA

CREATE

- Syntax

  CREATE COLUMN TABLE1 “VARIABLE1”.”VARIABLE2”
  {SUPPLIERID” INTEGER CS_INT NOT NULL ,

  “CITYID” INTEGER CS_INT,
  PRIMARY KEY (“SUPPLIERID”)}

- example of code is:

CREATE COLUMN TABLE “XTRA”.”DIMCUSTOMERV2”
{SUPPLIERID” INTEGER CS_INT NOT NULL ,
“CITYID” INTEGER CS_INT,
“COUNTRYID” INTEGER CS_INT,
“COMPANYNAME” NVARCHAR (20),
PRIMARY KEY (“SUPPLIERID”)}

Native SQL using SAP HANA

INSERT

- Syntax

  insert into table_name values { variable, field_name}

- example of code is:

```
insert into “XTRA” . “MYTESTTABLE” values
{
2,12345, ‘VW PASSAT’
};
```

Native SQL using SAP HANA

UPDATE

- Syntax

  update table_name set field_name = variable where condition

- example of code is:

```
update “XTRA” . “MYTESTTABLE”
set CARREGISTRATION = 12345
where CARID = 2
```

Native SQL using SAP HANA

DELETE

- Syntax
  delete from table_name where condition

- example of code is:

```
delete from “XTRA” . “MYTESTTABLE”
where CARID = 2
```

Native SQL using SAP HANA

JOIN

- Syntax

  select column1 from table_name1 inner join table_name2 on condition.

- example of code is:

```
select
T0. “COMPANYNAME”,
T1. “NETSALES”
from
“STS”. “DIMCUSTOMER” T0 inner join “STS”.”FCTCUSTOMERORDER” T1
on T0. “CUSTOMERID” = T1. “CUSTOMERID”
```

Native SQL using SAP HANA

SUB SELECT

- Syntax

  select column1 from table_name where (condition)
  having (condition1)
  (
  select (condition) from table_name1
)

Native SQL using SAP HANA

SUB SELECT

- example of code is:

```
SELECT COMPANYNAME AS COMPANY, ROUND(AVG (NETSALES),0) AS
AVERAGE_SALES
FROM “STS”. “DIMSUSTOMER”, “STS”. “FCTCUSTOMERORDER”
WHERE “STS”. “DIMCUSTOMER” = “STS”.”FCTCUSTOMERORDER”
GROUP BY CMPANYNAME
HAVING AVG (NETSALES) >
(
SELECT ROUND (AVG (NETSALES), 2) as AVERAGE
FROM “STS”. “FCTCUSTOMERORDER”
)
ORDER BY COMPANYNAME
```

Native SQL using SAP HANA

UNION

- Syntax

  select column1 from table1 UNION select column2 from table2

- example of code is:

```
{ SELECT COMPANYNAME AS COMPANY FROM “STS” . “DIMCUSTOMER”}
UNION
{ SELECT COMPANYNAME AS COMPANY FROM “XTRA” .
“ADDITIONALPROSPECTS”}
```

Native SQL using SAP HANA

DROP

- Syntax
  drop table table_name

- example of code is:

```
drop table STS.AAA ;

create table STS.AAA as
{
select *  from “STS”.”DIMCUSTOMER”
};
```

Native SQL using SAP HANA

VIEW

- Syntax

  create view view_name as

  select column1 column2 as field_name from table_name

- example of code is:

```
create view STS.STS_VIEW as
SELECT  countryname as COUNTRY,
SUM(netsales) as TOTAL_SALES
FROM "STSFLAT"."STSCUSTOMERFLATFILE"
GROUP BY countryname
HAVING sum(netsales) > 4000000
order by COUNTRYNAME
```

Native SQL using SAP HANA

SCHEMA

- Syntax
  create / drop schema schema_name

- example of code is:

create schema newuseradditionalschema owned by newuser

drop schema “NEWUSERADDITIONALSCHEMA”

Native SQL using SAP HANA

TABLE TYPES

- Syntax

  alter table table_name alter type row or column

- example of code is:

```
create column table sts.columnstoretable
(columna int)

alter table “STS”.”COLUMNSTORETABLE” alter type row;
```

Native SQL – Syntax check

Native SQL Statement Testing is done in SAP HANA Studio

As there is no syntax check for native SQL statements in ABAP, it can be very difficult to ensure the syntax is correct.

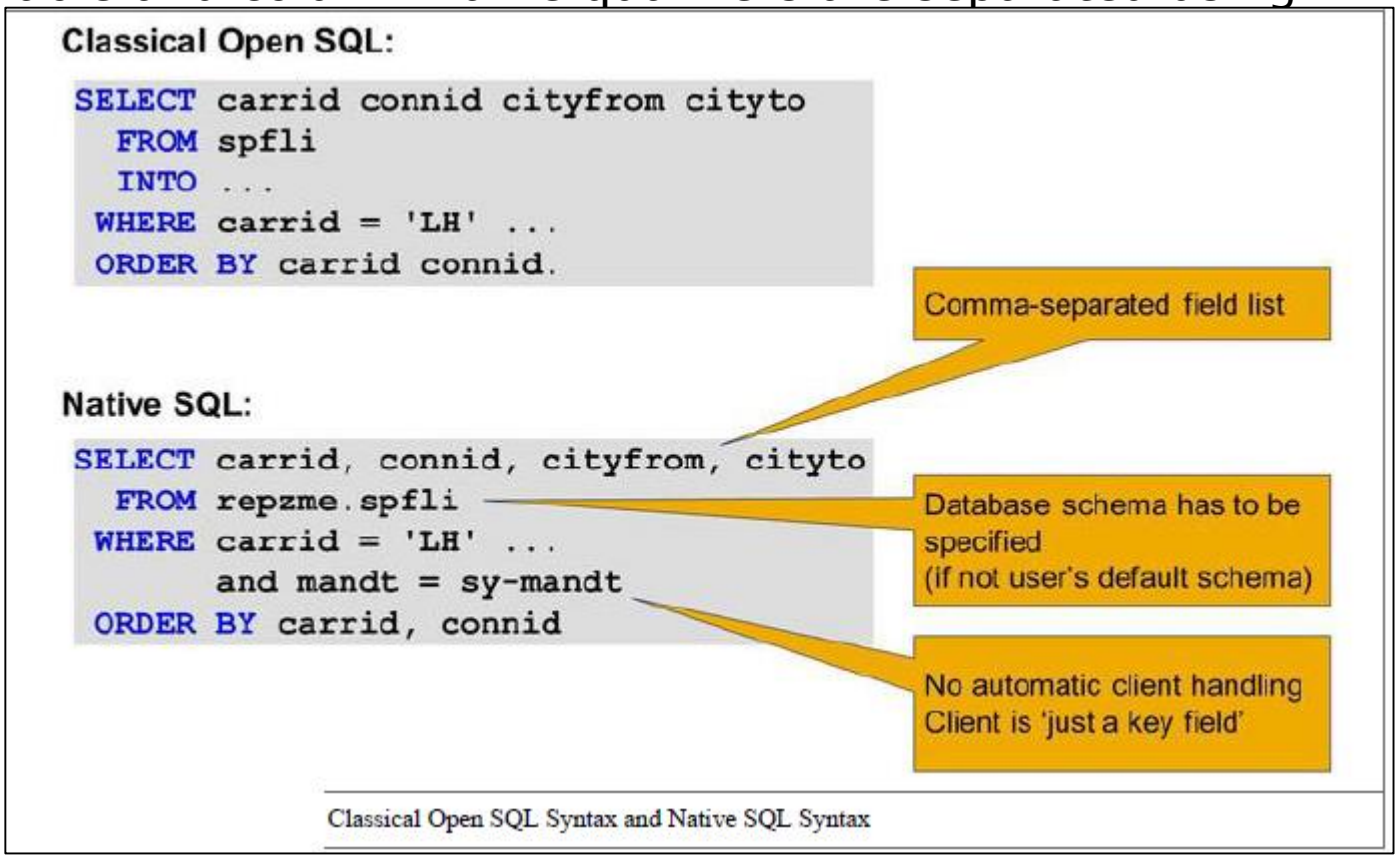
A convenient solution can be to use the SQL Console view of the

SAP HANA Studio.

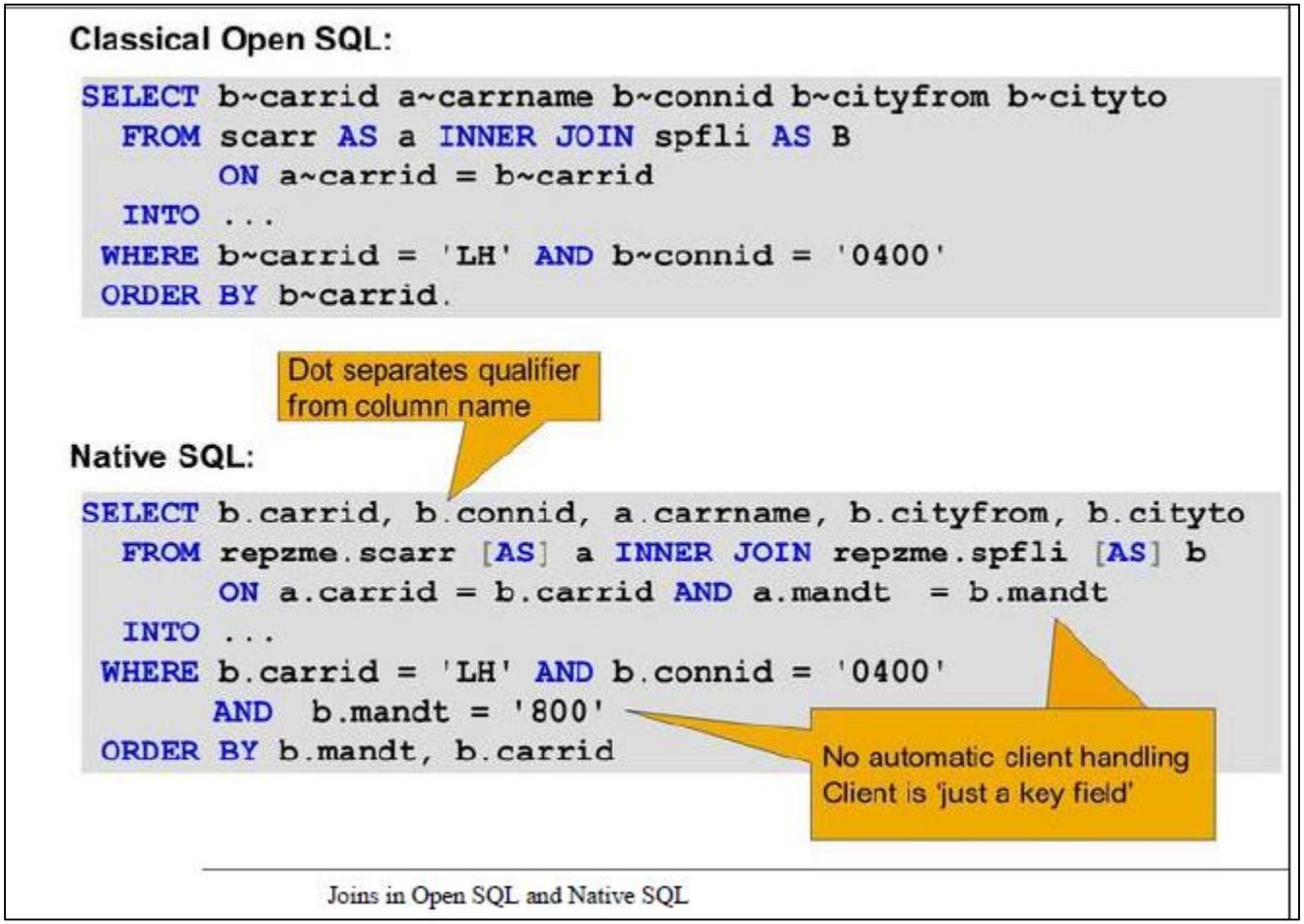
Difference between classical open SQL and Native SQL

Important Syntax Differences Between Native HANA SQL and Open SQL

- Column lists are comma separated
- Table and column name qualifiers are separated using “.”



Difference between classical open SQL and Native SQL



Summary

In this lesson, you have learnt:

- How to use native SQL for SAP HANA
- Different SQL syntaxes used for SAP HANA

Review Question

SQLScript is used in SAP HANA when other modeling constructs of HANA such as Attribute views or Analytic views are not sufficient.

- True
- False

Capgemini

Thank you