Applies to:
Web Intelligence BI 4.0 SP4 Rich Client
Information Design Tool 4.0 SP4
SAP HANA revision 34

Summary
The goal of this document is to present report design best practice use cases for SAP BI4 Web Intelligence connecting to SAP HANA for optimal performance.
In this best practice we will use a simple Web Intelligence report that will connect to an HANA Analytic View through a Universe.
We will also demonstrate how design mistakes can lead to performance degradations. This use case assumes that you already created your HANA Analytic Views and we will work only from Information Design Tool and Web Intelligence.

Authors: Romaric Sokhan, David Francois Gonzalez
Company: SAP
Created on: 27 July 2012
# Table of Contents

Goal of this document ................................................................. 3  
Summary .................................................................................... 3  
Target audience ................................................................. 3  
Prerequisites ........................................................................ 3  

Structure of the HANA Analytical View and Universe ......................... 4  
Analytic View modeled from VBRK table (Billing data) ....................... 4  
Universe modeled from the Analytic View .................................. 4  

Web Intelligence reporting on top of the Universe and Analytic View .......... 6  
Typical Web Intelligence Reporting using local microcube ................... 6  
Comments on this use case ...................................................... 8  

Optimization of the Web Intelligence report for SAP HANA .................. 9  
Stripping the query .................................................................... 9  
Implementing a drill solution to leverage HANA on-the-fly fast aggregation .................. 11  
Comments on this use case ...................................................... 15  

Recommendations for optimal Web Intelligence reporting on SAP HANA .......... 15  
Related Content ...................................................................... 16  
Copyright .............................................................................. 17
Goal of this document

Summary

The goal of this document is to present report design best practice use cases for SAP BI4 Web Intelligence connecting to SAP HANA for optimal performance.

In this best practice we will use a simple Web Intelligence report that will connect to an HANA Analytic View through a Universe. We will also show how design mistakes can lead to performance degradations. This use case assumes that you already created your HANA Analytic Views and we will work only from Information Design Tool and Web Intelligence.

Target audience

The targeted audience is BI analysts, BI Project Managers, Report and Universe developers who need to understand how to get the most of SAP HANA with Web Intelligence 4.0.

The reader must read the prerequisites below first, have at least a basic understanding of the universe technology and how to use Information Design Tool as well as how to create a simple Analytic View in SAP HANA Studio.

Prerequisites

Frequently Asked Questions about Web Intelligence on SAP HANA

Universes on SAP HANA : Best practices
http://scn.sap.com/docs/DOC-23256
http://scn.sap.com/docs/DOC-20569

HANA modeling Best practices
http://scn.sap.com/docs/DOC-22475
Structure of the HANA Analytical View and Universe

Analytic View modeled from VBRK table (Billing data)

The most optimal way to have good performance with SAP HANA is to create Analytic or Calculation View. In this use case, we will consume an Analytic View called AN_BILLING_LIST that refers to the VBRK table.

Please refer to the Universes and HANA modeling: Best practices if you need more details about HANA views (see prerequisites or related content).

Universe modeled from the Analytic View

The Universe designer simply drags and drops the Analytic View (AN_BILLING_LIST) from the _SYS_BIC schema into the Master view on the right hand side.
Once the Data Foundation is created, you need to create a Business Layer. We will need to create one Measure based on the `NETWR` field that we will rename to “Net Revenue”. This measure will use a SUM aggregation (in the SELECT statement) and aggregation function (for local cube calculation).

You can now export your Business Layer; this will convert your project into the UNX format that will be consumable by DSL-based client tools (Web Intelligence, Dashboards, Crystal for Enterprise).
Web Intelligence reporting on top of the Universe and Analytic View

Typical Web Intelligence Reporting using local microcube

Start Web Intelligence and select the Universe you just exported. You will be prompt by the Query Panel. Now select the Billing folder from the left hand pane, and drag and drop it into the Result Objects Pane as in the screenshot below.

Click on Run Query to generate the report with your query selection.

In the next page, we are showing the SQL statement sent by WebIntelligence to HANA:
SELECT SUM("_SYS_BIC"."i059086/AN_BILLING_LIST"."NETWR"),
"_SYS_BIC"."i059086/AN_BILLING_LIST"."WAERK",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."VTWEG",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKART",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKTYP",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."KUNNR",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."NAME1",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."CITY1",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."POST_CODE1",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."REGION",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."REGION.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKDAT_YEAR",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKDAT_QUARTER",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKDAT_MONTH",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKDAT_WEEK",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_TIME_CALQUARTER",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_TIME_CALMONTH",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_TIME_CALWEEK",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."DATE_SAP",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."DATE_SQL",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_DIVISION_SPART",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_DIVISION_SPART.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_DIVISION_MANDT",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_ORG_VKORG",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_ORG_VKORG.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_ORG_MANDT",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_ORG_WAERS",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_ORG_WAERS.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."BUKRS",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."BUKRS.description"
FROM "_SYS_BIC"."i059086/AN_BILLING_LIST"
GROUP BY "_SYS_BIC"."i059086/AN_BILLING_LIST"."WAERK",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."VTWEG",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKART",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKTYP",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."KUNNR",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."NAME1",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."CITY1",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."POST_CODE1",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."REGION",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."REGION.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKDAT_YEAR",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKDAT_QUARTER",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKDAT_MONTH",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."FKDAT_WEEK",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_TIME_CALQUARTER",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_TIME_CALMONTH",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_TIME_CALWEEK",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."DATE_SAP",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."DATE_SQL",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_DIVISION_SPART",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_DIVISION_SPART.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_DIVISION_MANDT",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_ORG_VKORG",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_ORG_VKORG.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_ORG_MANDT",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_ORG_WAERS",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."AT_SALES_ORG_WAERS.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."BUKRS",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."BUKRS.description"
Comments on this use case

As you can see in this use case, the amount of data retrieved in the local cube is 8911 rows and the time spent to execute and fetch data into the cube is 13 seconds (as a result of the large SQL statement sent to HANA).

Although the report displays with acceptable performance, that could worsen with other type of queries if the amount of rows is significantly higher. The benefit of this workflow is that as the data is now local, the user can perform further offline local analysis, offline calculation… (see Webi on HANA FAQ for more details). By avoiding sending queries, you remove some load off the HANA server.

The report above is **NOT** optimized for HANA. The report only displays *Country* and *Revenue*, which represents only 3 rows. We want to push all the calculation and aggregation down to HANA and retrieve only those 3 rows (next section)
Optimization of the Web Intelligence report for SAP HANA

Stripping the query

From the same report, go back to the Query Panel and remove all the dimensions that are not involved in the report, which means everything but Country Description and Net Revenue.

Here we simulated the “Query Stripping” feature that is currently available only for OLAP data sources. We simply removed the objects that were not involved in the initial report display.

The Query Panel should now look like this:

The generated SQL is now as follows:

```sql
SELECT
    SUM("_SYS_BIC"."i059086/AN_BILLING_LIST"."NETWR"),
    "_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY.description",  count(*)
FROM
    "_SYS_BIC"."i059086/AN_BILLING_LIST"
GROUP BY
    "_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY.description"
```
Click on *Run Query* again

The report now returns in 1 second and only 3 rows have been retrieved in the local microcube with the same exact results. All the aggregations have been pushed down to HANA and just the result set has been fetched (SQL statement is much smaller).

But now what if the user needs to analyze and see more data? We need to implement a drill down solution, in the next section.
Implementing a drill solution to leverage HANA on-the-fly fast aggregation

Go back into your Business Layer in IDT and select *Navigation Path*.

In the screenshot below, we created 2 customs *Navigation Path* that will define the direction of the user drill.

Once you defined your *Navigation Path*, export your Universe again and go back to the Web Intelligence report.

In your Webi report *Properties*, select the option “*Use Query Drill*” (for more information about this function please refer to the FAQ about Web Intelligence on SAP HANA)
In the Analysis tab, activate the *Drill*.

Note that only 1 dimension and 2 measures are available in the query.

Drill down on « USA », you will notice that the dimension « Region » has been added to the query.
Let's have a look at the SQL. We will notice that `REGION` has been added to the SELECT and a WHERE CLAUSE has been also added:

```sql
SELECT SUM("_SYS_BIC"."i059086/AN_BILLING_LIST"."NETWR"),
"_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."REGION.description"
FROM
"_SYS_BIC"."i059086/AN_BILLING_LIST"
WHERE
"_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY.description" = 'USA'
GROUP BY
"_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."REGION.description"
```

A further click on `Region` now adds the `City` dimension into the query:

---

**Number of rows in the local cube:** 4 rows

**Number of rows calculated in HANA:** 438,872 rows

**Time spent to calculate and retrieve the data:** 4 seconds

<table>
<thead>
<tr>
<th>City</th>
<th>Net Revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Los Altos</td>
<td>231,686,864.98</td>
</tr>
<tr>
<td>Los Angeles</td>
<td>248,406,713.29</td>
</tr>
<tr>
<td>Palo Alto</td>
<td>10,72</td>
</tr>
<tr>
<td>San Jose</td>
<td>194,814,657.12</td>
</tr>
</tbody>
</table>
The query has now been updated as in the below screenshot

And translates into the following SQL:

```sql
SELECT SUM("_SYS_BIC"."i059086/AN_BILLING_LIST"."NETWR"),
"_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."REGION.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."CITY1"
FROM
"_SYS_BIC"."i059086/AN_BILLING_LIST"
WHERE
("_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY.description" = 'USA'
AND
"_SYS_BIC"."i059086/AN_BILLING_LIST"."REGION.description" = 'California')
GROUP BY
"_SYS_BIC"."i059086/AN_BILLING_LIST"."COUNTRY.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."REGION.description",
"_SYS_BIC"."i059086/AN_BILLING_LIST"."CITY1"
```

**Note:** With *Query on Drill* activated, if the user drills up, he will notice that the dimension not needed anymore will be removed from the left panel, thus from the query.
Comments on this use case

As you can see with the above screenshots, by drilling down or up into the data, WebIntelligence narrows down the amount of data to be analyzed by HANA and queries only the data needed for the report. Each user drill action results to a sub-second response time query. In a scenario with hundreds of millions of records, the performance is even more obvious compared to traditional databases, which would need the creation of materialized tables in order to achieve good performance.

Recommendations for optimal WebIntelligence reporting on SAP HANA

- Always retrieve the least possible quantity of data in your client tool using filters or restricted list of values. Query only the data that is needed.

- Use the Web Intelligence local microcube if you need to perform offline analysis and calculations but be aware that this could impact performance depending on the volume of data retrieved.

- Retrieve aggregate data, not detailed, whenever possible.

- Retrieve only the columns you need: HANA being a columnar database, if you don't ask for a column, the system will not spend any time on it.

- Build your universe to avoid the need of multiple queries on the same schema from the client tool. ‘Query on query’ sub-query functionality could be avoided by table joins.

- In Web Intelligence prefer to use the query-drill mode (drill online) rather than to retrieve all columns in the document (also called Scope of analysis). You’ll be able to drill down and up in the document and retrieve the needed amount of data by automatically filtering the query when drilling.

- Make sure the SQL query sent by Web Intelligence is executing fast in HANA studio. If not, rework your HANA models.
Related Content

Frequently Asked Questions about Web Intelligence on SAP HANA

Universes on SAP HANA : Best practices
http://scn.sap.com/docs/DOC-23256
http://scn.sap.com/docs/DOC-20569

HANA modeling Best practices
http://scn.sap.com/docs/DOC-22475