# Setup of Star Schema Benchmark

Use SSB to load M7/S7 system to demo benefit of DAX.

* Install Oracle DB S/W
  + 12.1.0.2 (or 12.2 ?)
  + Bundle patch (req. for 12.1!)
  + Create DB on ASM (preferably) (SID should be orcl to match my lab)
* Create SSB data and load DB
  + Install “electrum” and build dbgen
  + Create (bigfile+autoextend) tablespace and user “ssb” (to later have an easy way to delete everything)
  + Create tables for customer, lineorder, …
  + Generate data in flat files, use parallelization for lineorder
  + Run catldr.sql
  + Load data into DB using “sqlldr direct=true”

alter table … memcompress for query high priority critical

* Prepare and run SSB
  + Extract some supplier id and mfg locations from tables into .csv file
  + Install jmeter, install extension plugin
    - Install combined graph plugin
    - Install stepped load plugin
  + Setup jmeter
    - Create query in jmeter
      * Use join
      * Cycle through .csv created before
    - Step load from 10 to 100 “users”

## Installation of Oracle Software and Patches

This recipe will detail the installation of 12.1.0.2 and the corresponding Grid Infrastructure, which will be used for ASM setup. Installing 12.2 should work as well. Search on edelivery.oracle.com for “database 12” and “Grid Infrastructure”, pick versions 12.1.0.2 and platform Solaris SPARC 64bit for both. Now you can download the zip files:

* The database is V46108-01.zip , -02.zip, in total 2.7GB
* Grid Infrastructure in V46109-01.zip , -02.zip, in total 2GB

The bundle patch can be downloaded from MOS, support.oracle.com. As with most Oracle patches you first need the latest OPatch binary, which has the generic patch number 6880880, and get the latest bundle patch, January 2017 PSU would be 24733088.

My testbed is a LDOM on a T7-4 system made up from four internal disks, one SPARC M7 CPU and all its local memory, around 256GB in my case. The system is running the latest Solaris 11.3 SRU, 18 at the time writing.

To provide enough shared memory space for the database edit the limits for the group staff in /etc/project:

group.staff:10::::\

project.max-shm-memory=(privileged,250000000000,deny)

Create a user for Oracle:

* Name it “oracle” like the rest of the world does
* Copy the zip files into his home directory
* Create a mountpoint /u01 with enough space to hold the Oracle software except for the database files, logs etc (which will go into ASM later)

Unpack all zip files, I like to keep them in a directory “software” to keep the home directory clean. The zip files unpack into a directory called “database” and one called “grid”.

Now install Grid Infrastructure **first** by executing runInstaller from the “grid” directory, accept all defaults. Then install the database in the same way.

Make the oracle user owner of the disks that shall be under ASM control, and then invoke asmca from $GRIDHOME/bin (should be something like /u01/app/12.1.0/grid/bin) Its poit&click, create a diskgroup called “ASM” containing the disks. Now you’re done with ASM.

Create a database with the SID “orcl”, leave the storage destination at its default ASM.

Now first unpack the OPatch patch right into $ORACLE\_HOME and $GRIDHOME, if you are cautious first rename the existing OPatch directory and then unpack. Become root user, unpack the bundle patch and then apply it by executing $GRIDHOME/OPatch/opatchauto apply <patch nr>

Now you are done with the initial install of the database, these steps are not specific for Star Schema, you will have to go through these steps for most Oracle database installs.

## Creation of Star Schema Data and Database Load

To run queries of at least a little relevance to real life one should have data to run queries against, and a common way of getting data is to generate them. Star Schema is no exception to that rule, and since it is a derivative of TPC-H it uses a similar data generator. My recipe uses the electrum dbgen, available from github:

https://github.com/electrum/ssb-dbgen

Click “clone/download” and unpack the zip file (it unpacks into a directory of its own. It builds straightforward using gcc, I would recommend to not waste time to build it with something else or do optimization. The runtime of the actual data generation step is irrelevant to the test here. So you want to have gcc in your search path, type “make” to build the “dbgen” binary in the unpack directory.

Now run the actual data generation step, I did it right in the build directory. The important decision you have to make is “how much data?” The size is controlled by the so called “scale” factor. As a rule of thumb, scale 100 is a bit more than 100GB of data (when it has been loaded into the database), so I make sure that I have twice the disk space to be able to create the data to be loaded on disk and later actually load it.

Five tables are to be created: customer, date, supplier, parts, and lineorder, which is the largest and it is mainly its size which is controlled by scale. (Scale 100 results in about 600Mio rows in lineorder):

* ./dbgen –T c –s 100 creates a file “customer.tbl”containing artificial customer contact data
* ./dbgen –T d –s 100 creates “date.tbl”
* ./dbgen –T p –s 100 creates “parts.tbl”
* ./dbgen –T s –s 100 creates “supplier.tbl”
* ./dbgen –T l –C 32 –s 100 creates in parallel 32 files “lineorder.tbl.n” which later will all be loaded into the database (If you have enough resources you can increase the degree or parallelism here. Parallelization does pay off here because of the size of lineorder, the others generate quickly single threaded)

The next step is the actual load of the tables we will use Oracle’s sqlldr to load the clear text files into the database.

But first the actual tables have to be generated.

## Load Tables into Database

Loading is best done with SQL Loader, when loading the lineorder table make sure your tablespace is large enough and index maintenance is turned off (the bulk of data resides in lineorder table!):

sqlldr userid=ssb/ssb \

direct=true skip\_index\_maintenance=true \

data=<name of dbgen output file> \

control=<name of control file>

The control files are all pretty straightforward, e.g. the one for the customer table should look like this:

load data

into table customer

append

fields terminated by '|'

(c\_custkey, c\_name, c\_address, c\_city, c\_nation, c\_region, c\_phone, c\_mktsegment)

## Setup Benchmark / Load Generator

Download jmeter from apache.org, any binary should do. Edit the “jmeter” script and replace all occurrences of awk with gawk (The script assumes certain behavior of awk, which Solaris’ awk does not have. Maybe you need to install gawk)

Start jmeter and change “Look&Feel” to “Metal” or something else other than the default Darcula (which doesn’t render properly at least in my setups)