Index Lab

1. Create the person_tbl as an exact copy of the pers_person_tbl.
2. Identify the primary key on the person_tbl and create a primary key.
3. Create the svcmbr_tbl as an exact copy of the per_svcmbr_tbl.
4. Identify the primary key on the svcmbr_tbl and create a primary key.
5. A. Create a join condition between the svcmbr_tbl and the person_tbl with the following
   Information: Display ssn_sm, rec_prec, name_ind, states_us, cum_ret_pt, gr_abbr_code
   For the report. Save the query as join_tbl.sql
   B. How many rows are displayed?
   C. After a good execution, look at the execution plan using either SQL*PLUS autotrace, or explain plan or use SQL Developer graphical interface to display execution plan. Save the
   Execution plan to a file call exec1.html in a directory with your name on the root drive (/home/oracle or C:)
   D. Now filter the query by just retrieving the same data but only those soldiers whose social
   Security numbers are less than 100551212. Save this to a file called join_tbl2.sql
   E. Display the execution plan. Is it different from the previous plan? Save this information to a
   File called exec2.html in the same directory as used earlier.
   F. Why is it different?
6. A. Create a foreign key on the svcmbr_tbl to create Referential Integrity between the
   Person_tbl and svcmbr_tbl.
   B. Rerun the join_tbl.sql script created at 5A.
   C. Look at the execution plan. Is it different than any of the previous plans? Save this execution plan as exec3.html
   D. Rerun join_tbl2. Again review the execution plan for differences. Save the execution plan as exec4.html
7. A. Create a new script using the join_tbl.sql script called join_index.sql and add a where clause which looks like this:
   Where states_us in ('TX', 'NY','CA').
   B. Look at the execution plan. What indexes if any does it use? Save this plan as exec5.html
8. Create a non-unique index on the states_us column using the person_tbl.
9. Run the join_index.sql script and using the explain plan of SQL*PLUS or SQL Developer.
   Are there any differences in the plan? Why? Save this plan as exec6.html
10. Create the unit_tbl as an exact copy of the pers_unit_tbl.

11. Identify the primary key on the unit_tbl and create the primary key.
12. Create a script which requires the following information: mpc, asg_seq_code, ssn_sm, rec_prec, cum_ret_pt, gr_abbr_code, upc, uname and zip code. Name this query join_unit.sql
13. Run this script to display the plan. Save this plan to a file called exec7.html
14. Create the foreign key on the svcmbr_tbl to create referential integrity between the unit_tbl
    And the svcmbr_tbl.
15. A. Run a script which requires the following information:
    Ssn_sm, gr_abbr_code, cum_ret_pt on the svcmbr_tbl. Name this script soldier1.sql.
16. Look at the execution plan. Does it use an index? Save it as exec8.html
17. Create a composite index composed of ssn_sm, gr_abbr_code and cum_ret_pt.
18. Create statistics for the svcmbr_tbl by executing the dbms_stats utility which would generate
    Statistics for both tables and indexes.
19. Rerun the soldier1.sql script and display the execution plan. Is it different from the prior execution plan? Save the execution plan as exec9.html
20. A. Create and run a script which contains the following info: ssn_sm, rec_prec, name_ind, states_us, dob from the person_tbl. Bring all soldiers back who were born before 01-Jan-1990. Save this script as cdates.sql.
    B. Produce an execution plan and identify how it runs. Save this output as exec10.html.
21. A. Create a partitioned table called soldier_date which puts all soldiers born before 01-Jan- 1990
    Into a partition called ELDER which is in a tablespace called ELDER which is in a directory located on root (/home/oracle or C:) called ELDER. You will need to create this tablespace. The file name will be ELDER01.dbf and 100 megabytes. Allow it to autoextend by 2 mg when it runs out of space.
    B. Create another partition call MIDDLE which has all soldiers born between 01-Jan-1990 and 01-Jan-2000 and place that data in a tablespace called MIDDLE which is in a file located on root (/home/oracle or C:) in a directory called MIDDLE. This tablespace will need to be created also. The file name will be MIDDLE01.dbf and 100 megabytes. Allow it to extend by 2 mg when it starts to run out of space.
    C. Create another partition call MIDDLE which has all soldiers born between 01-Jan-1990 and 31-Dec-1999 and place that data in a tablespace called MIDDLE which is in a file located on root (/home/oracle or C:) in a directory called MIDDLE. The file name will be MIDDLE01.dbf and 100 megabytes. Allow it to extend by 2 mg when it starts to run out of space.
    D. The last partition will be called CURRENT and will be composed of all soldiers born after 01-JAN-2000. Place that data in a tablespace called CURRENT which is in a file located on root (/home/oracle or C:) in a directory called CURRENT.

The file name will be CURRENT01.dbf and 100 megabytes in size. Allow it to extend by 2 mg when it starts to run out of space.

22. Rerun the scripts cdates.sql Look at the plan, how does it differ from previous plans?
23. A.Let's create a bitmap index on a column in the person_tbl. The candidate columns are: sex, martl_stat or zip. Which column has the lowest cardinality? Run a query which uses the lowest cardinality column. Now let's make the person_tbl bigger. First, drop the person_tbl. Now run
    The imp command from a terminal or dos prompt.
    C:\ora12clabs> imp sidpers/password -- use the person_tbl export.dmp file.
    Then run the following command.
    B. SQL> select ssn_sm, rec_prec, name_ind, states_us, sex
    From person_tbl
    Where sex = 'F'
    Evaluate the plan produced.
    C. Create a bitmap on that column called bitmap_index1.
    SQL> create bitmap index bitmap_index1 on person_tbl(sex);
    D. Now evaluate the new plan produced.

24. Create an IOT from the person table called person_iot which includes the above data (ssn_sm, rec_prec,name_ind,states_us,dob and apft_score) with the primary key of ssn_sm and rec_prec and sends eth_gp, martl_stat, race_pop_gp, rel_denom, sex, hgt_ind, wt_ind, loc_data_pers to an overflow tablespace called user_data.

25. Create another IOT from the svcmbr table called svcmbr_iot which includes the mpc, asg_seq_nbr, ssn_sm, rec_prec, upc,gr_abbr_code, cum_ret_pt and date_rec_stat with the primary key of mpc and asg_seq_nbr while sending the following columns (date_asgn_loss_rsn, org_ident, expir_date_ing, date_init_procrmt,retn_wvr, date_mand,rem, date_end_eval_period, afqt_score_gps) to an overflow tablespace called user_data.

26**.** Create a data cluster called the SVCMBR_UNIT_CLUS which has cluster key composed of UPC of varchar2(5). Create a cluster index called SVCMBR_UNIT_CLUSX on the cluster. Create and load the cluster with a table called UNIT_CLUS which is a copy of the UNIT_TBL into the cluster and add a second table into the cluster called SVCMBR_CLUS which is a copy of the SVCMBR_TBL. Review the difference in the data between the SVCMBR_TBL and the SVCMBR_CLUS table.