**Group 2**

**Date: 19-04-2018**

**DBA625 Group Assignment 2**

*Prof. Nebojsa Conkic*

**Completed By:**

Avinash Singh

Gabriel Appugliese

Eric Ferguson

Yik Chee Wong

**PART : ONE**

**a) Create new user called DAVE and assign RED and TEMP tablespaces to him, also DEFAULT profile. He should be allowed to use only 2M of RED tablespace, and also 1M of INDX tablespace.**

**CREATE USER DAVE**

**IDENTIFIED BY abcd**

**DEFAULT TABLESPACE RED**

**QUOTA 2M ON RED**

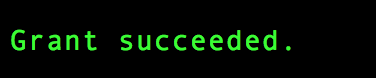
**TEMPORARY TABLESPACE TEMP**

**QUOTA 1M ON INDX**

**PROFILE DEFAULT;**

****

**GRANT CONNECT, CREATE TABLE to DAVE;**

****

**b) Create another user called LARA and assign BLUE and TEMP tablespaces to her, also DEFAULT profile, but no quota initially for her and her account should be locked. Both users should get role that will allow them to connect to SQL and individual privilege for creating tables.**

**CREATE USER LARA**

**IDENTIFIED BY ABCD**

**PROFILE DEFAULT**

**PASSWORD EXPIRE**

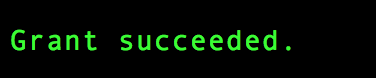
**ACCOUNT LOCK**

**DEFAULT TABLESPACE BLUE**

**TEMPORARY TABLESPACE TEMP;**

****

**GRANT CONNECT, CREATE TABLE to LARA;**

****

**c) Create new profile called ITPROF so that:**

**• One session per user is allowed**

**• Idle CPU time is no more than 1 hour**

**• Three false logins are allowed and that account should remain locked for 2 minutes after the fourth false login**

**CREATE PROFILE ITPROF**

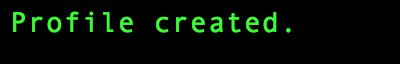
**LIMIT**

**IDLE\_TIME 60**

**SESSIONS\_PER\_USER 1**

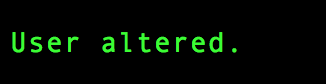
**FAILED\_LOGIN\_ATTEMPTS 3**

**PASSWORD\_LOCK\_TIME 2/1440;**

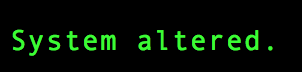
****

**d) Assign this profile now to DAVE and enable all restrictions for profile ITPROF. Then connect to SQL\*PLUS as DAVE and try to connect to SQL\*PLUS as DAVE again (in a second session). What happened?**

**ALTER USER DAVE PROFILE ITPROF;**

****

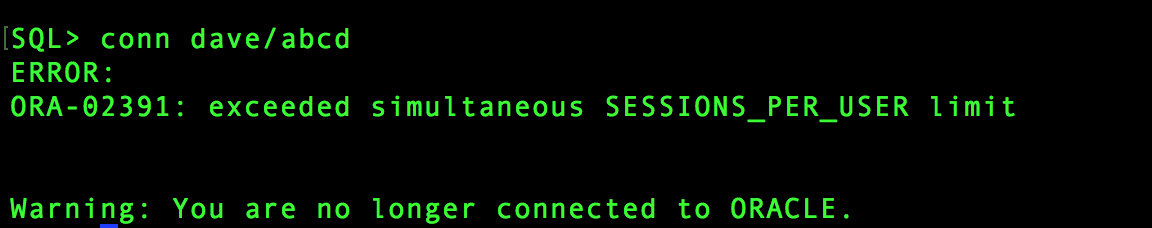
**ALTER SYSTEM SET resource\_limit=true;**

****

**CONN DAVE/abcd**

****

**CONN DAVE/abcd**

****

**e) As SYSTEM modify profile ITPROF so that only two false logins are allowed and that password life time is only 2 months with the grace period of 5 days. Then try to login as DAVE with the wrong password 3 times. Wait only 1 minute and provide now the right password. What happened and how can you rectify this problem à show both methods?**

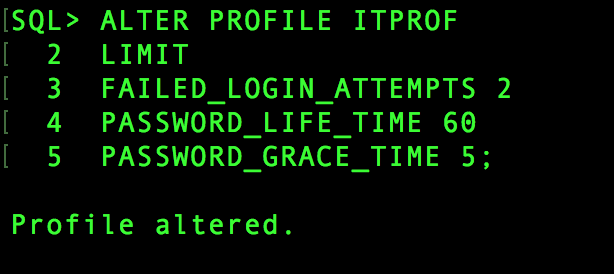
**ALTER PROFILE ITPROF**

**LIMIT**

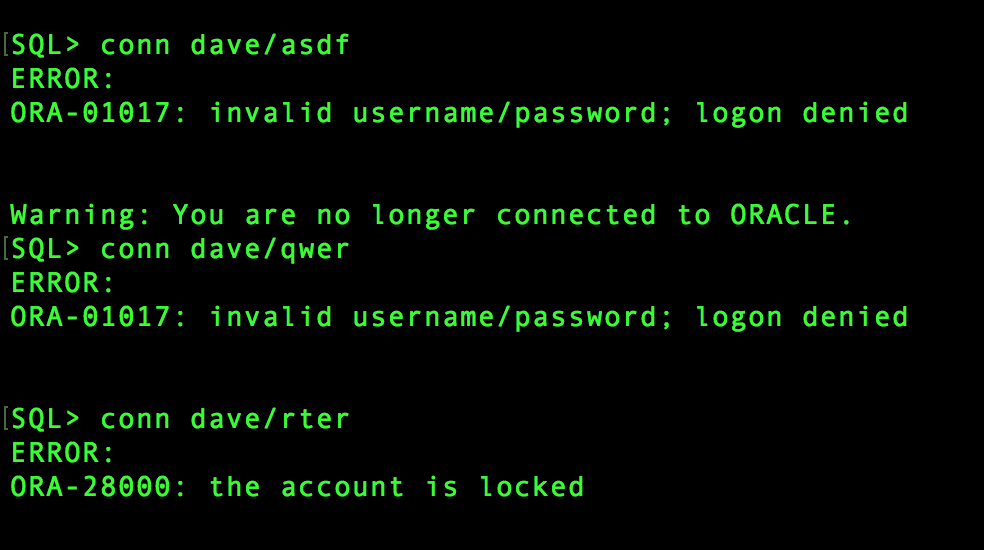
**FAILED\_LOGIN\_ATTEMPTS 2**

**PASSWORD\_LIFE\_TIME 60**

**PASSWORD\_GRACE\_TIME 5;**

****

**CONNECTING AS DAVE with 3 WRONG PWASSWORDS.**

****

**f) As SYSTEM try to create replica of SCOTT’s table EMP in both DAVE’s and LARA’s accounts. Was it successful in both cases? How can you fix this problem? After doing that, verify that LARA (after login) may access her table EMP.**

**CREATE TABLE LARA.EMP (**

**EMPNO NUMBER(4) NOT NULL,**

**ENAME VARCHAR2(10),**

**JOB VARCHAR2(9),**

**MGR NUMBER(4),**

**HIREDATE DATE,**

**SAL NUMBER(7,2),**

**COMM NUMBER(7,2),**

**DEPTNO NUMBER(2)**

**);**

**CREATE TABLE DAVE.EMP (**

**EMPNO NUMBER(4) NOT NULL,**

**ENAME VARCHAR2(10),**

**JOB VARCHAR2(9),**

**MGR NUMBER(4),**

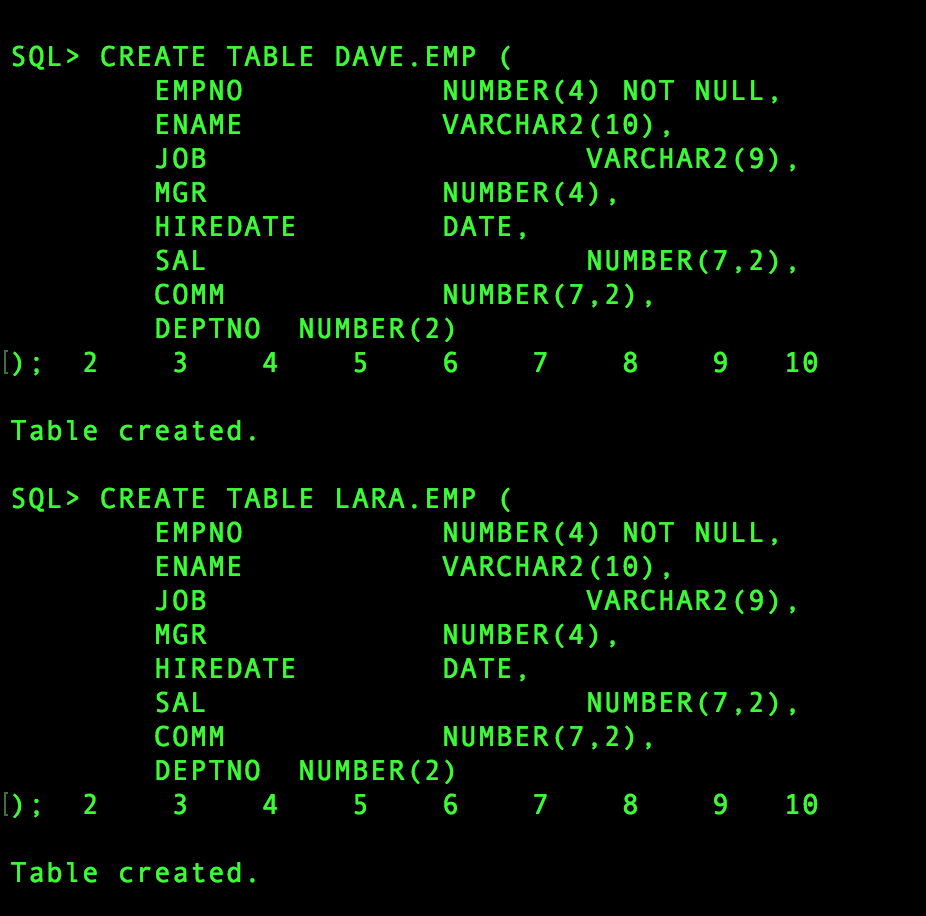
**HIREDATE DATE,**

**SAL NUMBER(7,2),**

**COMM NUMBER(7,2),**

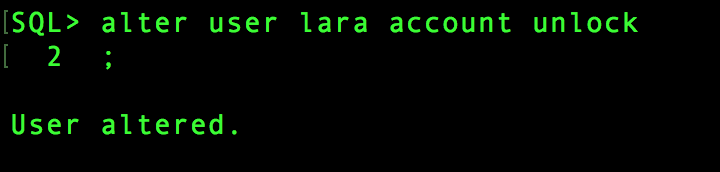
**DEPTNO NUMBER(2)**

**);**

****

**We need to unlock user LARA to be able to access EMP table.**

**\*\*I think I need to give some QUOTA to LARA in TABLE SPACES.**

****

**g) By joining 2 dictionary views display for users DAVE and LARA their account status, when the account will expire, what profile and tablespaces are assigned and what the current and**

**maximal byte situation is for these tablespaces?**

**SELECT U.USERNAME,**

**U.ACCOUNT\_STATUS,**

**U.EXPIRY\_DATE,**

**U.PROFILE,**

**Q.TABLESPACE\_NAME,**

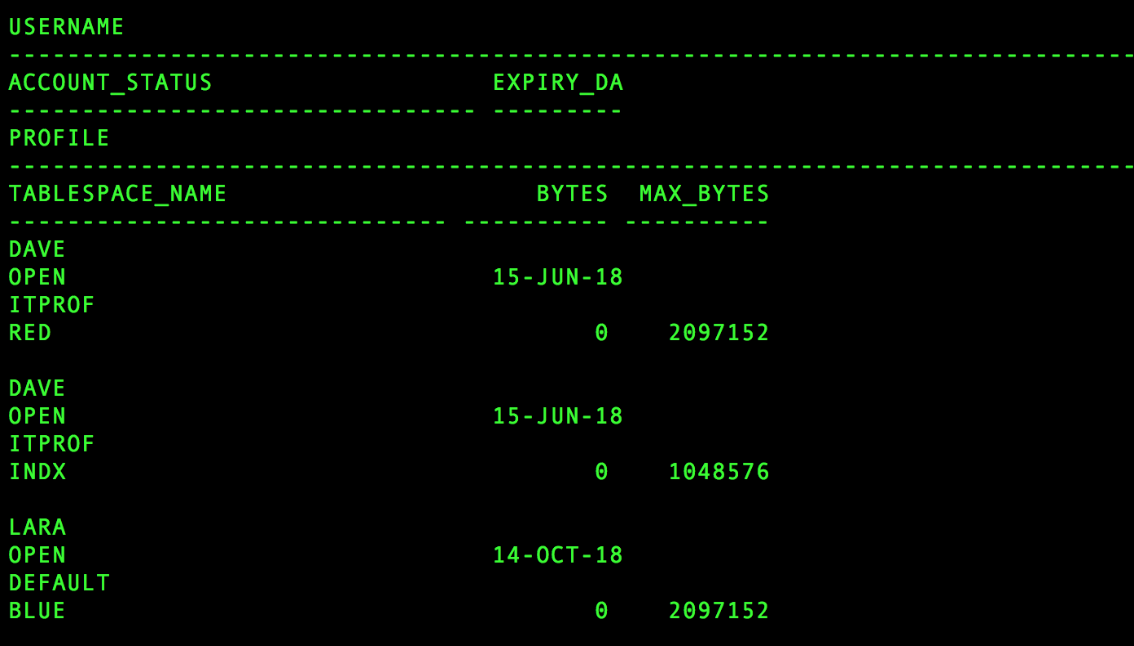
**Q.BYTES,**

**Q.MAX\_BYTES**

**FROM DBA\_USERS U, DBA\_TS\_QUOTAS Q**

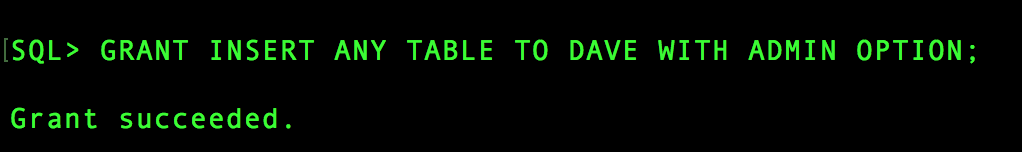
**WHERE U.USERNAME = Q.USERNAME**

**AND U.USERNAME IN ('DAVE','LARA');**

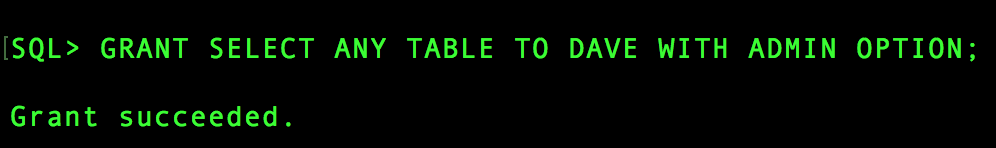
****

**h) Now give individual privilege to DAVE, so that he can browse and add data in a table in any account and also that he can continue to give this privilege. Then connect as DAVE and give the same privilege to LARA?**

**GRANT INSERT ANY TABLE TO DAVE WITH ADMIN OPTION;**

****

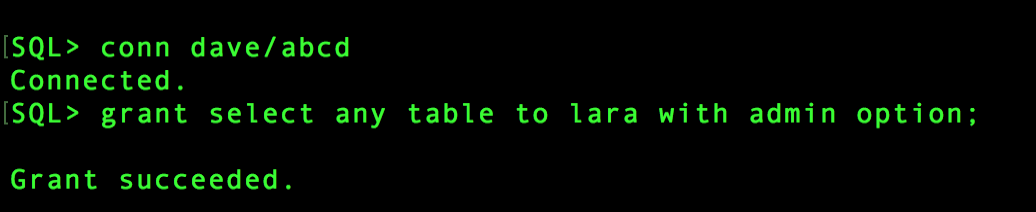
**GRANT SELECT ANY TABLE TO DAVE WITH ADMIN OPTION;**

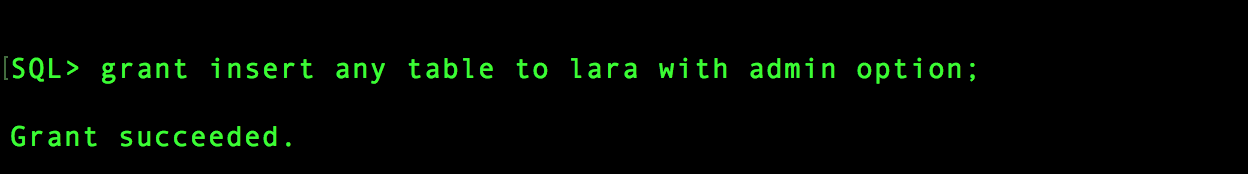
****

**CONN DAVE/abcd**

**GRANT SELECT ANY TABLE TO LARA WITH ADMIN OPTION;**

**GRANT INSERT ANY TABLE TO LARA WITH ADMIN OPTION;**

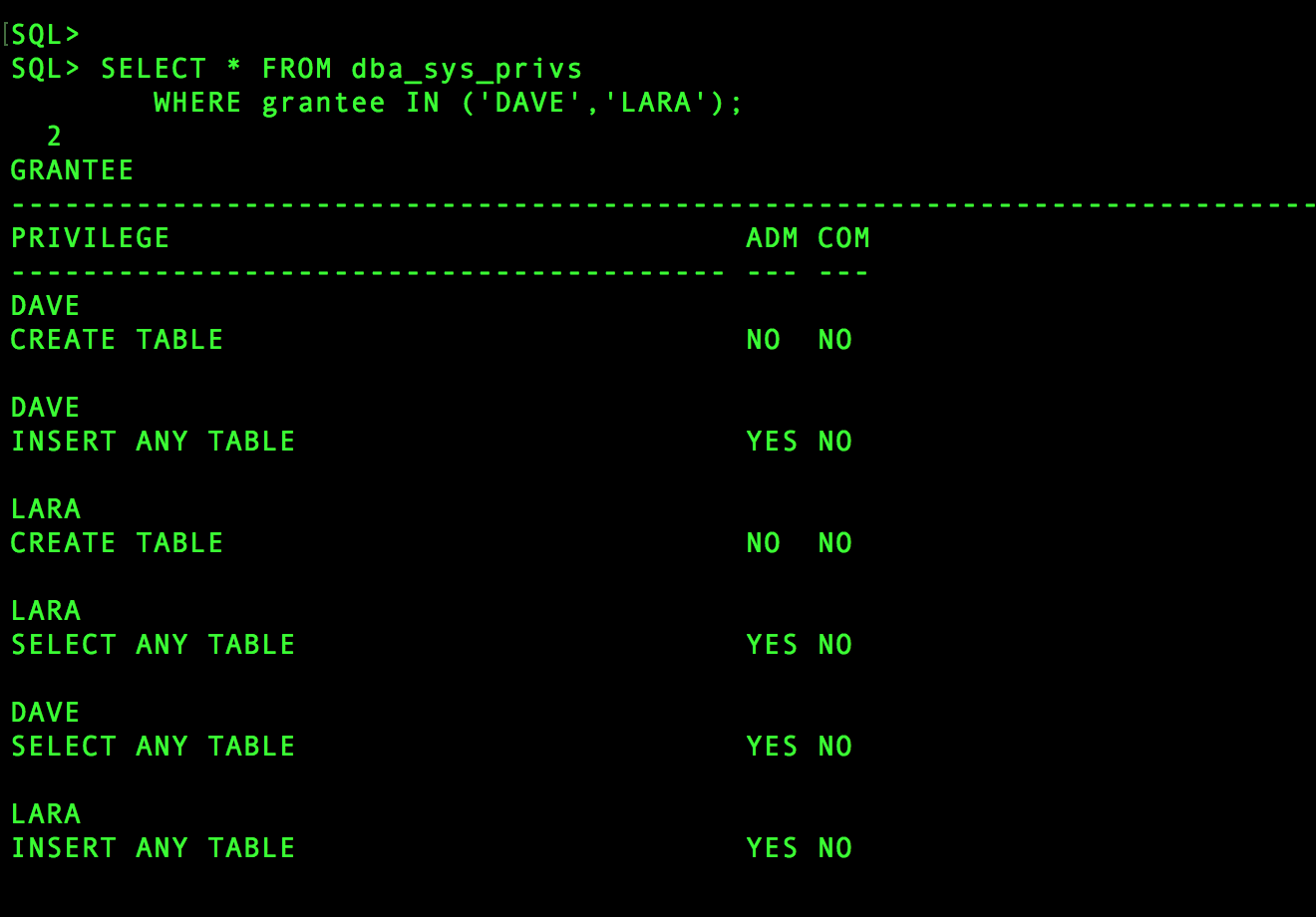
****

****

**i) As SYSTEM, check the appropriate dictionary view and observe only system privileges for those two users (show only relevant columns from this view).**

**SELECT \* FROM dba\_sys\_privs**

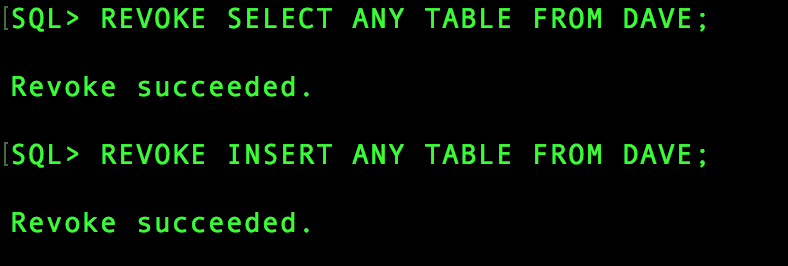
**WHERE grantee IN ('DAVE','LARA');**

****

**j) Remove the privilege given to DAVE in h), then connect as LARA and try to ADD one row into SCOTT’s table EMP. Was it successful and why? Then repeat step i) and explain what is different now?**

**REVOKE SELECT ANY TABLE FROM DAVE;**

**REVOKE INSERT ANY TABLE FROM DAVE;**

****

**INSERT INTO SCOTT.EMP (**

**EMPNO,**

**ENAME,**

**JOB,**

**MGR,**

**HIREDATE,**

**SAL,**

**COMM,**

**DEPTNO)**

**VALUES(**

**6199,**

**'Avinash',**

**'Developer',**

**7902,**

**'11-APR-1993',**

**60000,**

**800,**

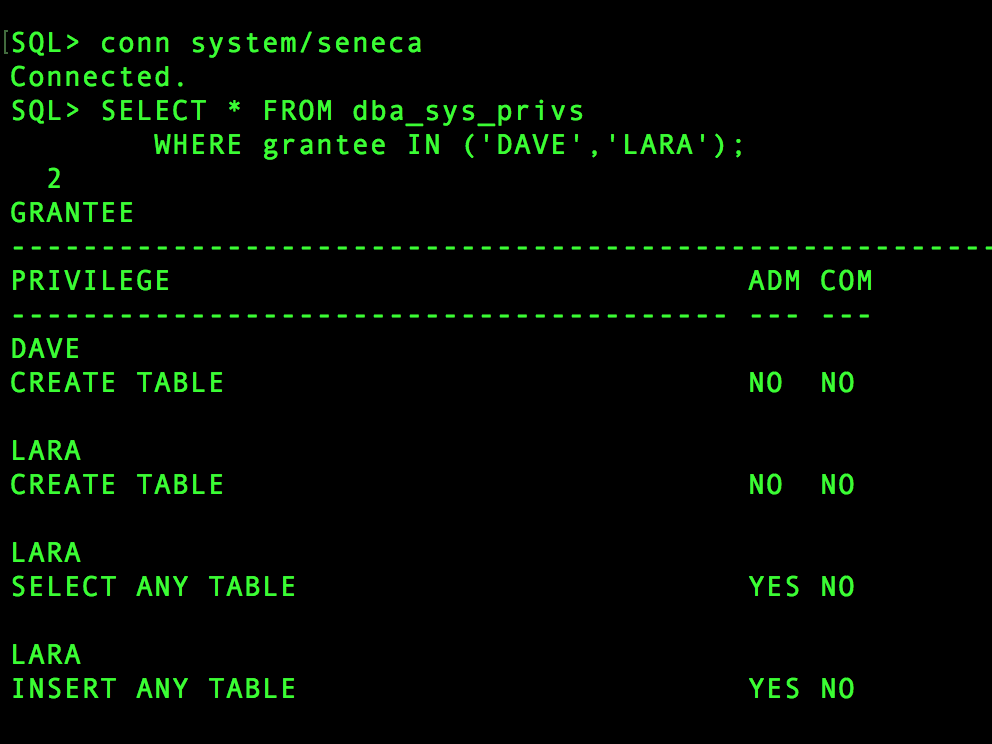
**20);**

****

**Why : Beacause we have revoked privilege from DAVE’s account without CASCADE option, So LARA still has the privileges.**

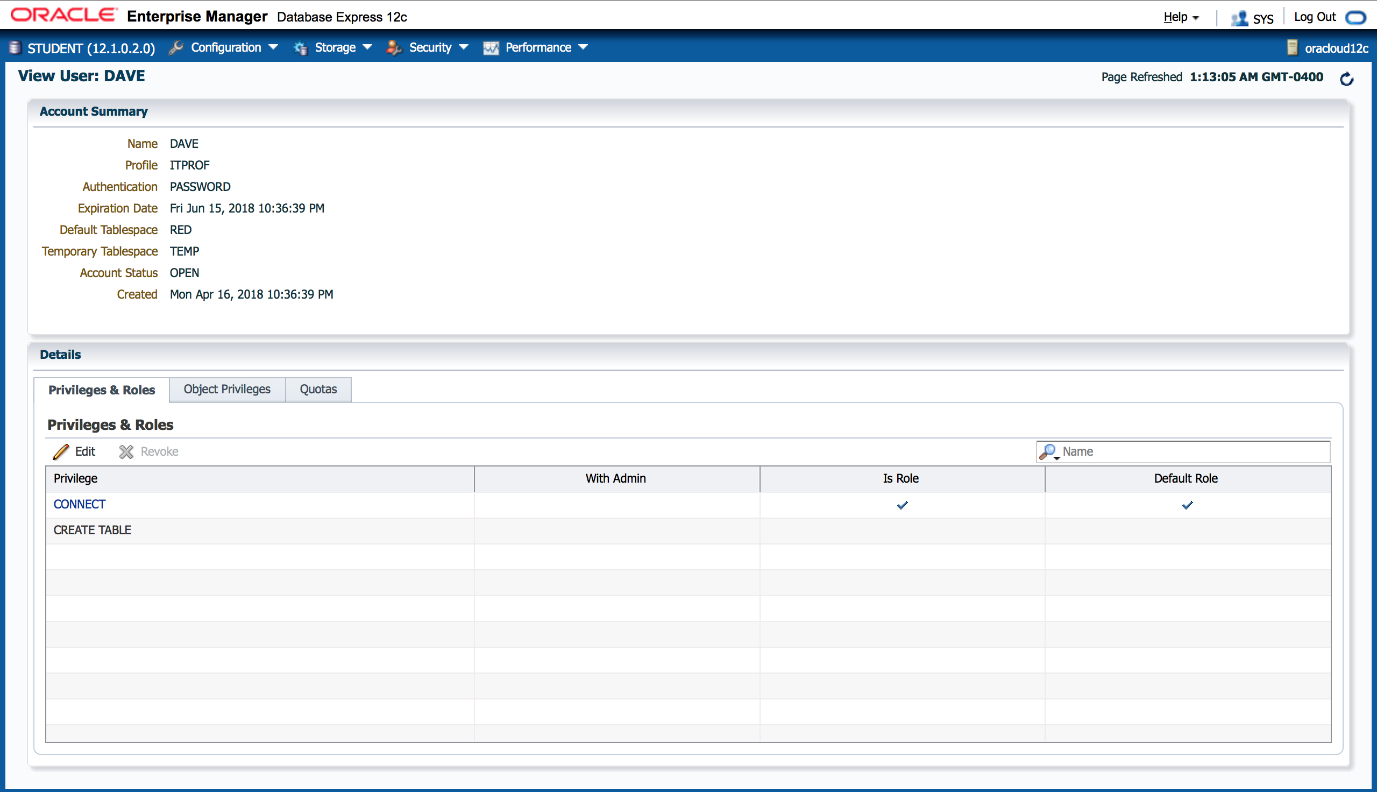
**SELECT \* FROM dba\_sys\_privs**

**WHERE grantee IN ('DAVE','LARA');**

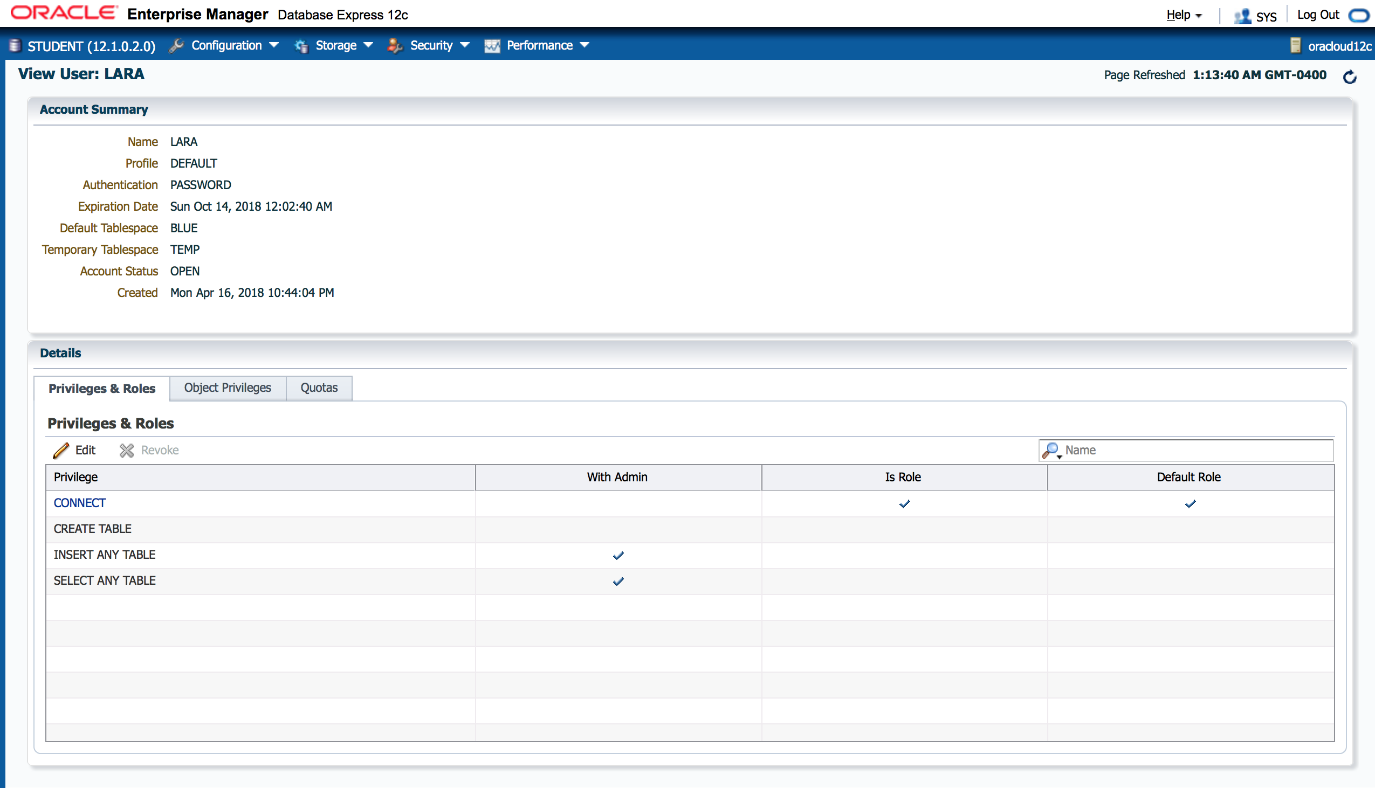
****

**k) In DB Express display situation for DAVE’s account firstly and then for LARA’s account as two different pages.**

**DAVE**

****

**LARA:**

****

**PART : TWO**

**Firstly run the script cr\_orders.sql as user DAVE (you will need to adjust this script regarding Tablespace name here)**

**RUN**

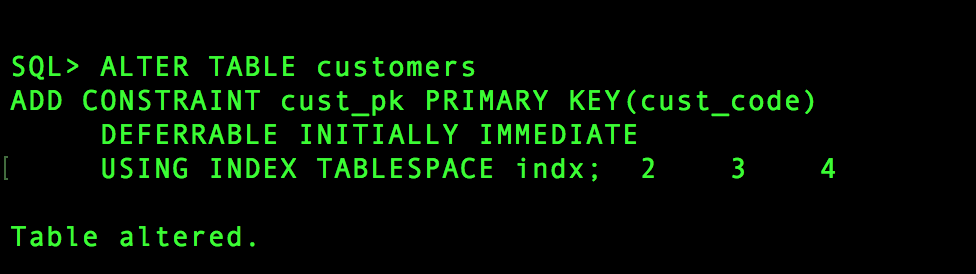
**a) As DAVE, create a PK constraint for table CUSTOMERS, so that its checking can be delayed (till saving) LATER in the future, but for now it will behave like the non-delayed one. The state should be set to check only incoming data. Also add PK constraint for table ORDERS, so that its checking can be delayed PROMPTLY after its creation, while its state should be set to check both existing and incoming data. Related indexes should go to Tablespace INDX.**

**ALTER TABLE customers**

**ADD CONSTRAINT cust\_pk PRIMARY KEY(cust\_code)**

**DEFERRABLE INITIALLY IMMEDIATE**

**USING INDEX TABLESPACE indx;**

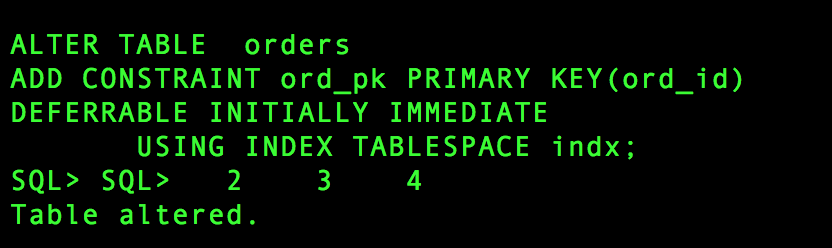
****

**ALTER TABLE orders**

**ADD CONSTRAINT ord\_pk PRIMARY KEY(ord\_id)**

**DEFERRABLE INITIALLY IMMEDIATE**

**USING INDEX TABLESPACE indx;**

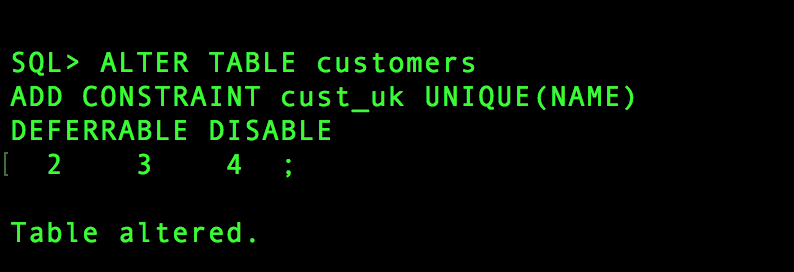
****

**b) Also add an UK constraint on column NAME in table CUSTOMERS, so that will have default value for either its state or mode, and sub-default for the other one. Think hard here. You may want to check question g) below in order to make a right choice. The related index will also go to tablespace INDX.**

**ALTER TABLE customers**

**ADD CONSTRAINT cust\_uk UNIQUE(NAME)**

**DEFERRABLE DISABLE**

****

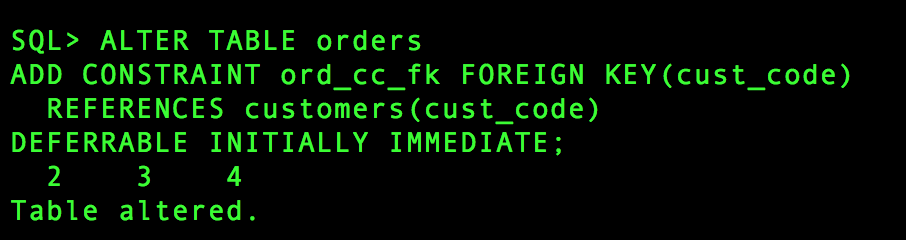
**c) Create a FK constraint for table ORDERS so that might be LATER manually delayed (at save time). This constraint state follows the default value.**

**ALTER TABLE orders**

**ADD CONSTRAINT ord\_cc\_fk FOREIGN KEY(cust\_code)**

**REFERENCES customers(cust\_code)**

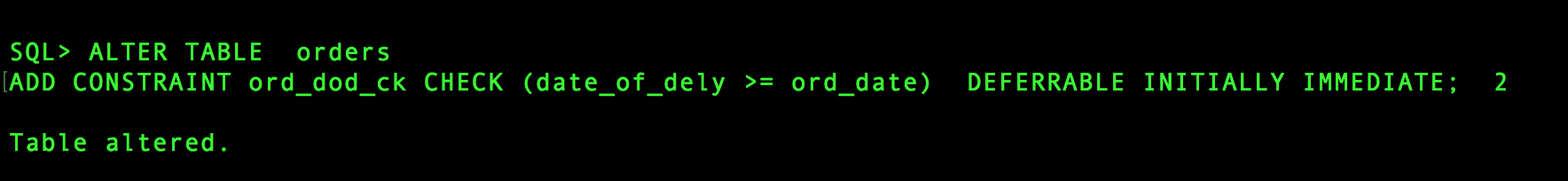
**DEFERRABLE INITIALLY IMMEDIATE;**

****

**d) Then declare a CK constraint with condition that date of delivery may not be later than two weeks from the order date and also not before it. This constraint's mode is the default one, while the state follows the sub-default value.**

**ALTER TABLE orders**

**ADD CONSTRAINT ord\_dod\_ck CHECK (date\_of\_dely >= ord\_date) DEFERRABLE INITIALLY IMMEDIATE;**

****

**e) As SYSTEM, join two most important constraint dictionary views to display the following: constraint name, type, status, validation, can be deferred or not, is it currently deferred, table name and what column(s) are involved for tables CUSTOMERS and ORDERS in DAVE’s account.**

**SELECT a.constraint\_name,**

**a.constraint\_type,**

**a.status,**

**a.deferrable,**

**a.deferred,**

**a.validated,**

**a.table\_name,**

**c.column\_name**

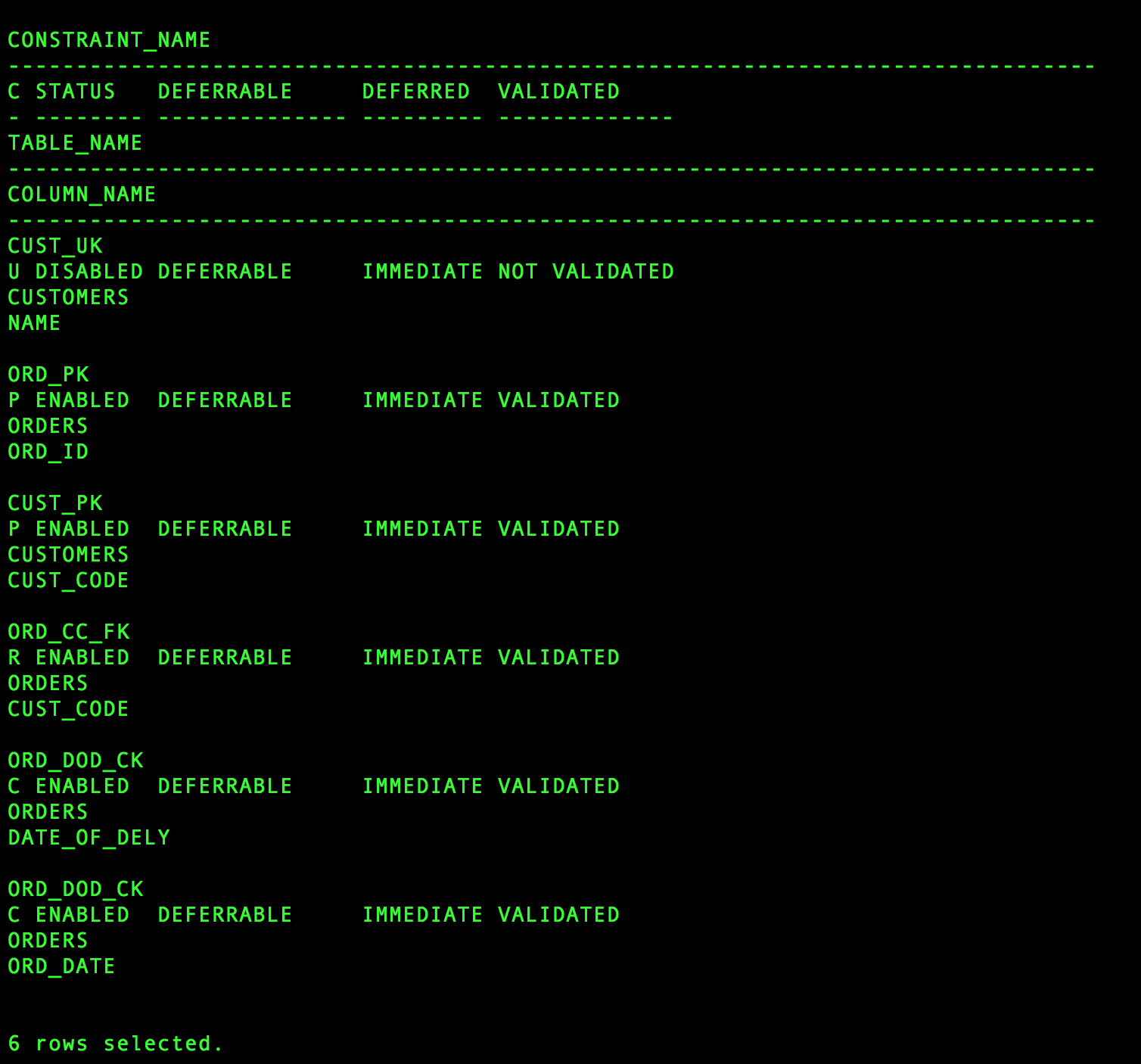
**FROM dba\_constraints a, dba\_cons\_columns c**

**WHERE a.constraint\_name = c.constraint\_name**

**AND a.table\_name IN ('CUSTOMERS','ORDERS')**

**AND a.owner = 'DAVE'**

**AND c.owner = 'DAVE';**

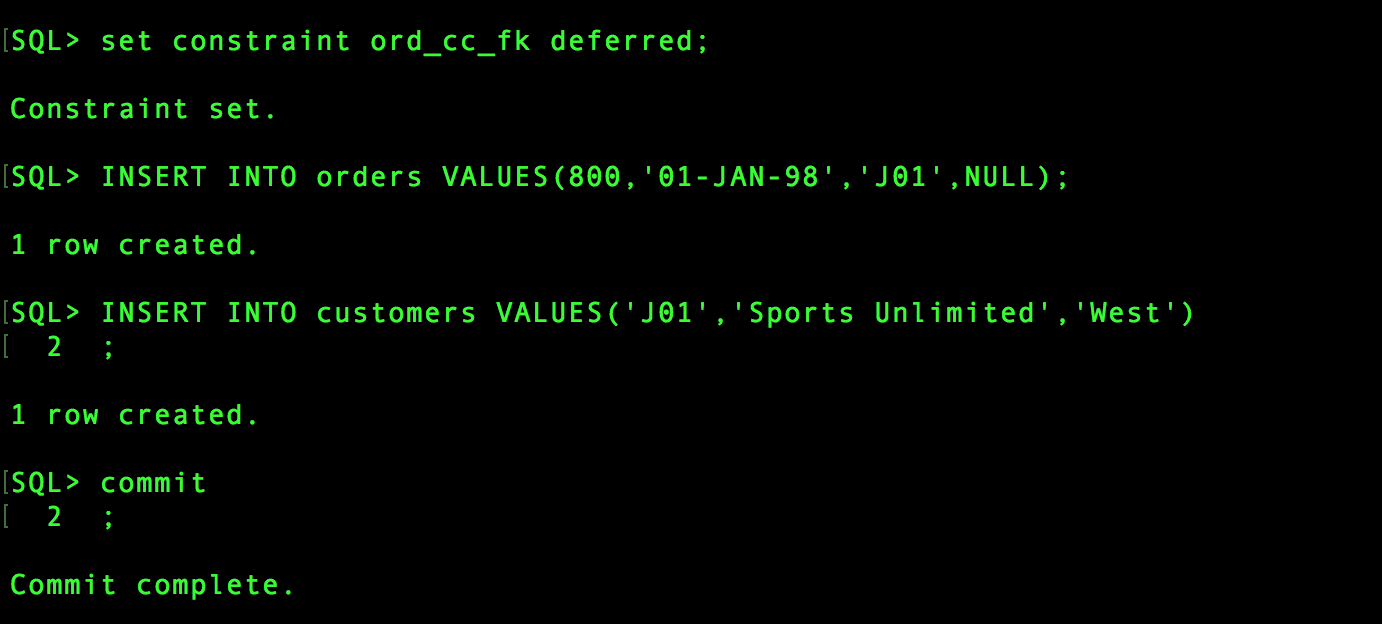
****

**f) As DAVE, try to make a short DML script (yours and original), that will show how it is possible to insert a child record before its parent record, if the child FK constraint checking is delayed till save time. Manually perform the change that will allow this check delay.**

**SET CONSTRAINT ORD\_CC\_FK DEFERRED;**

**INSERT INTO customers VALUES('J01','Sports Unlimited','West')**

**INSERT INTO orders VALUES(800,'01-JAN-98','J01',NULL)**

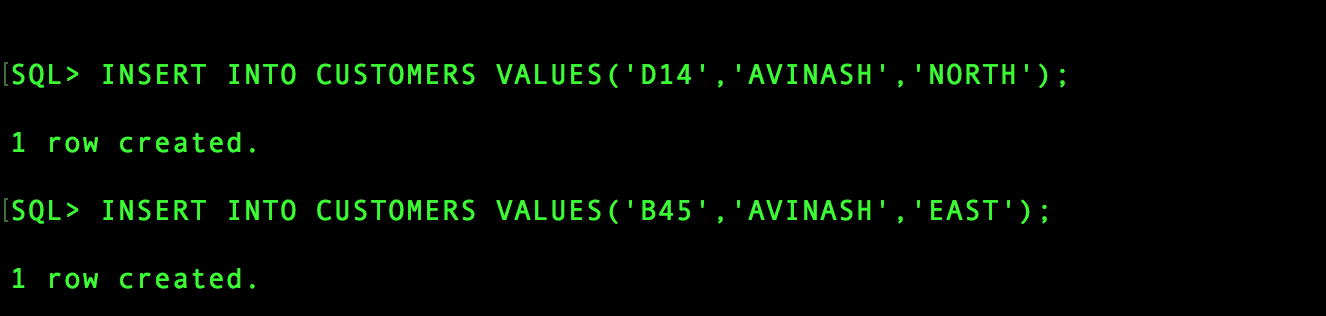
****

**g) Then make another short script that will show how it is possible to enter same names for different customers (three), if the appropriate constraint is turned off at the creation time.**

**SET CONSTRAINT CUST\_UK DEFERRED**

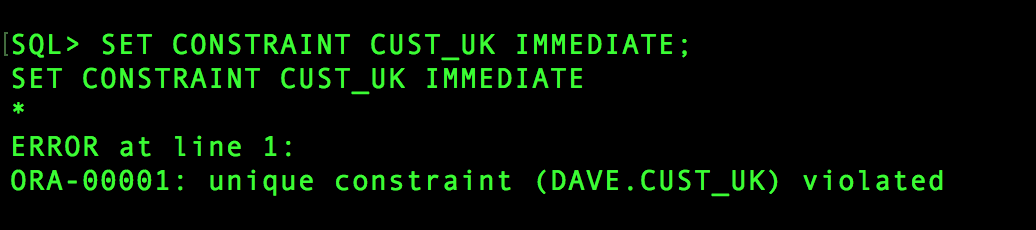
**INSERT INTO CUSTOMERS VALUES(‘D14’,’AVINASH’,’NORTH);**

**INSERT INTO CUSTOMERS VALUES(‘D15’,’AVINASH’,EAST);**

****

**h) Now try to set the UK constraint in DAVE’s table CUSTOMERS to the default state. What happened?**

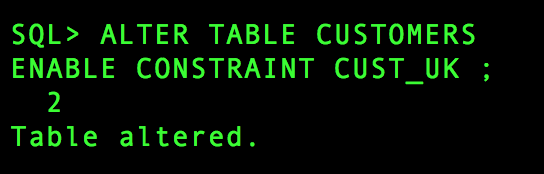
**SET CONSTRAINT CUST\_UK IMMEDIATE;**

****

**i) Then enable it so that it does NOT check the existing data. If not possible, fix this obstacle by creating a new object in DAVE’s schema.**

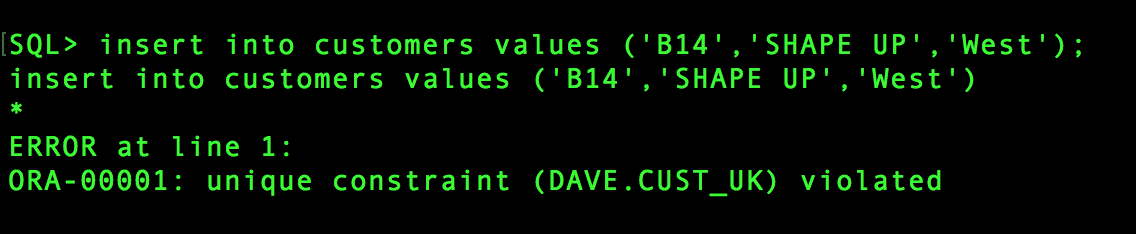
**ALTER TABLE CUSTOMERS**

**ENABLE CONSTRAINT CUST\_UK ;**

****

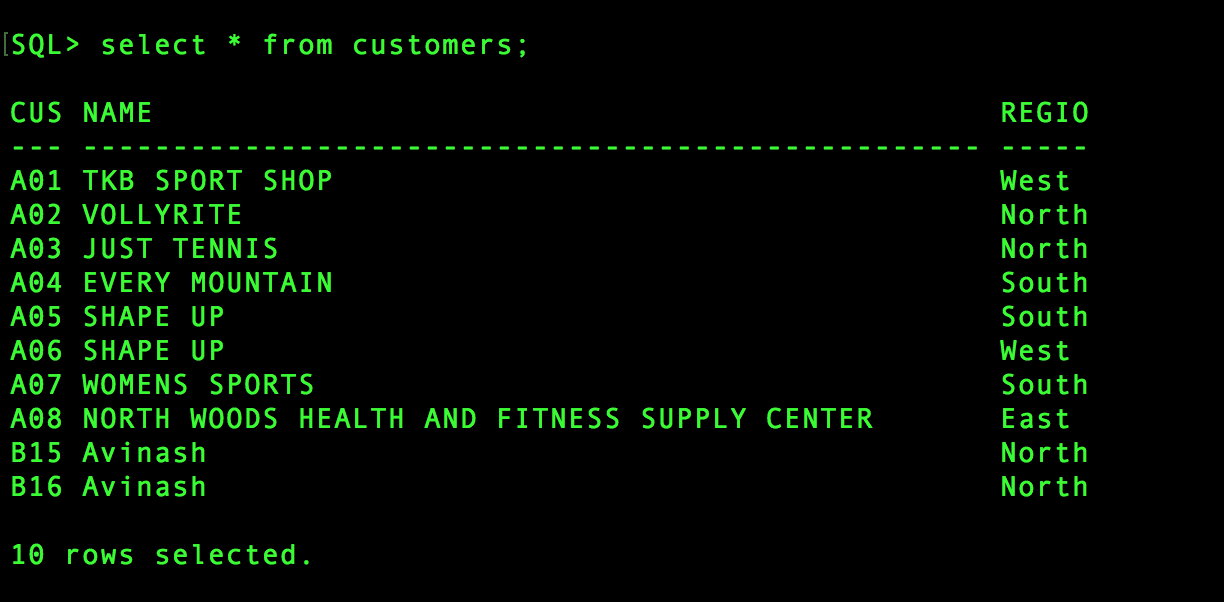
**j) Next, try to enter a row with an existing customer name. What happened?**

**INSERT INTO CUSTOMERS VALUES(‘B14’,’SHAPE UP’,West’);**

****

**k) Perform the five step recipe for consolidating the situation with the Name column in DAVE’s CUSTOMERS table (get rid of all customers with duplicate names by modifying duplicates).**

**STEPS:**

**1. Inserting two duplicate values in customer table – B15 and B16 which has NAME column as Unique key.**

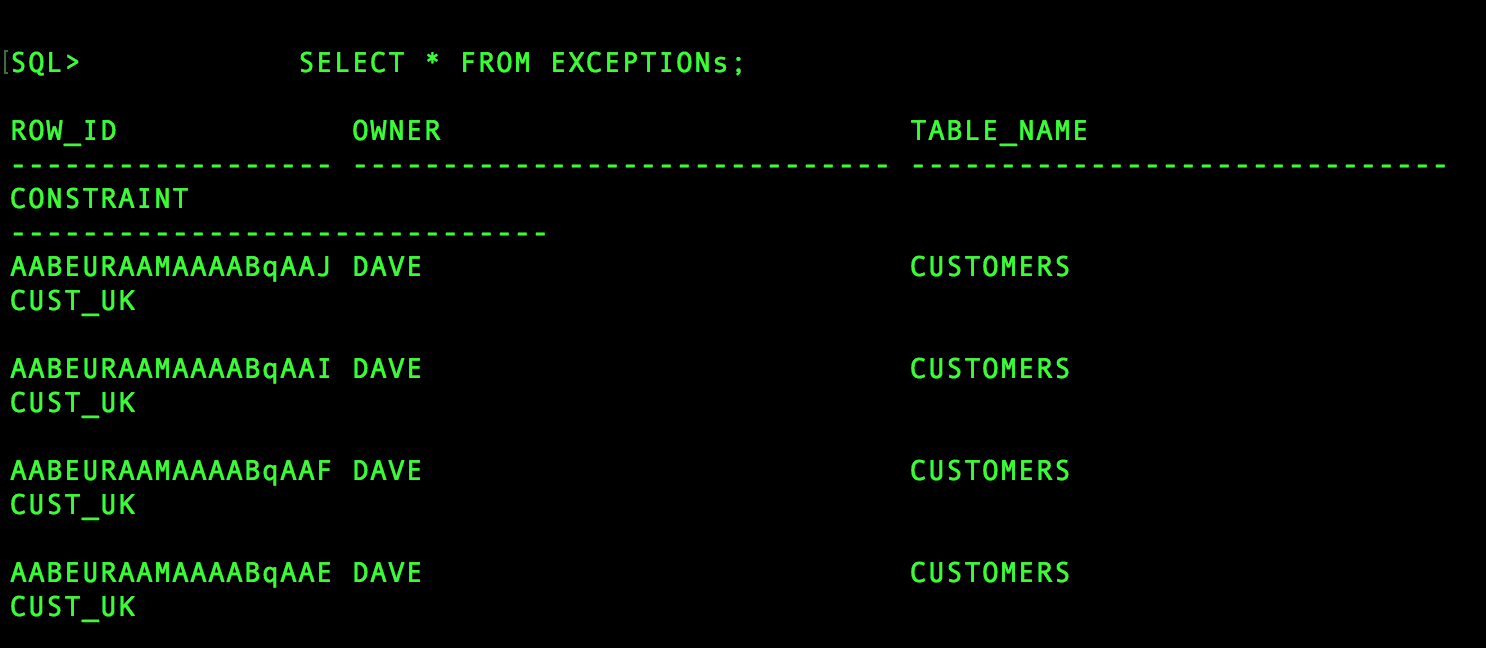
**2.** **ALTER TABLE customers**

**ENABLE CONSTRAINT cust\_uk**

**EXCEPTIONS INTO exceptions;**

****

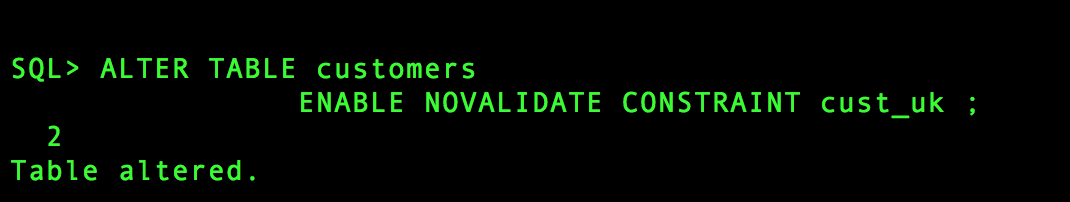
**SELECT \* FROM EXCEPTION;**

****

**3.**

**ALTER TABLE customers**

**ENABLE NOVALIDATE CONSTRAINT cust\_uk ;**

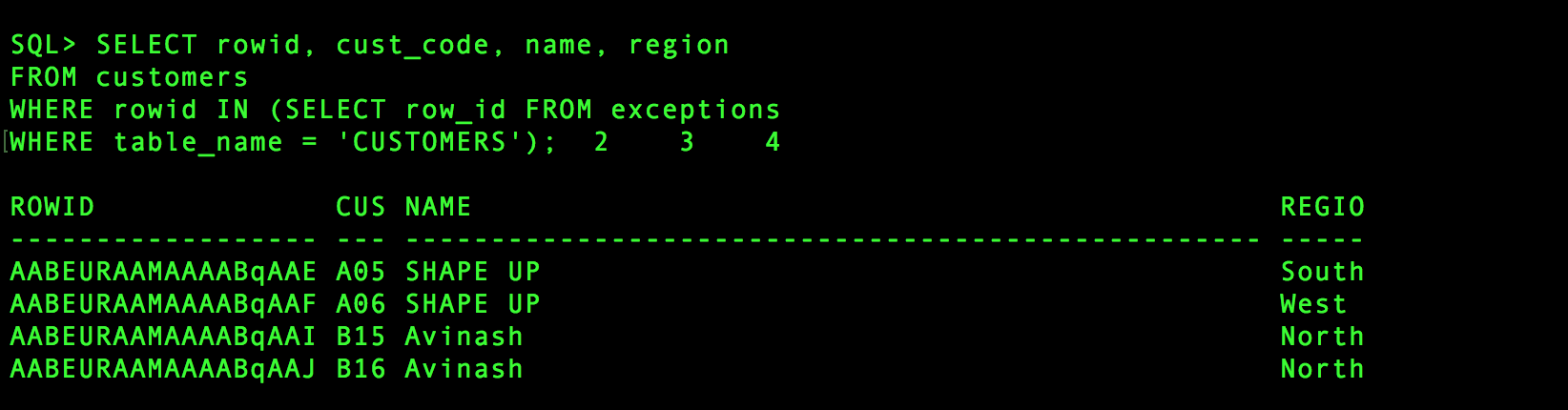
****

**4. SELECT rowid, cust\_code, name, region**

**FROM customers**

**WHERE rowid IN (SELECT row\_id FROM exceptions**

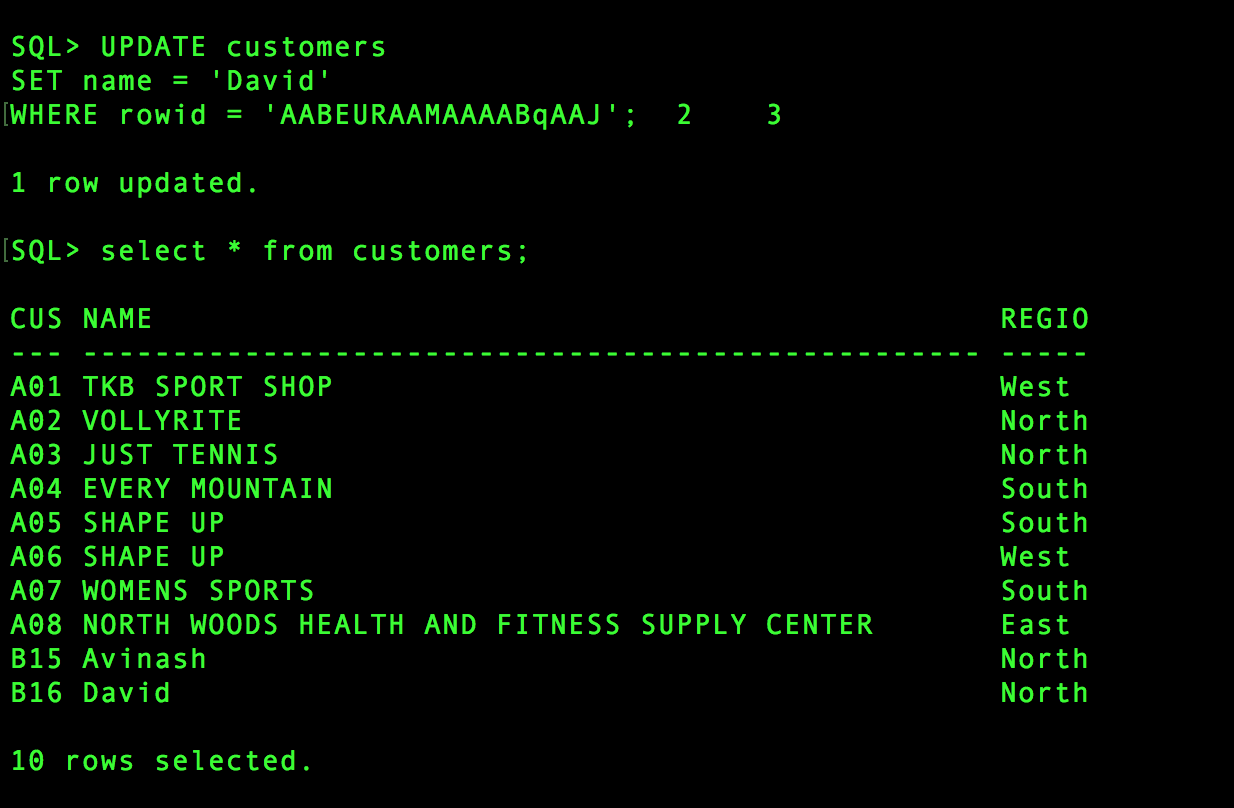
**WHERE table\_name = 'CUSTOMERS');**

****

**5. UPDATE customers**

**SET name = 'David'**

**WHERE rowid = 'AABEURAAMAAAABqAAJ';**

****

**Three:**

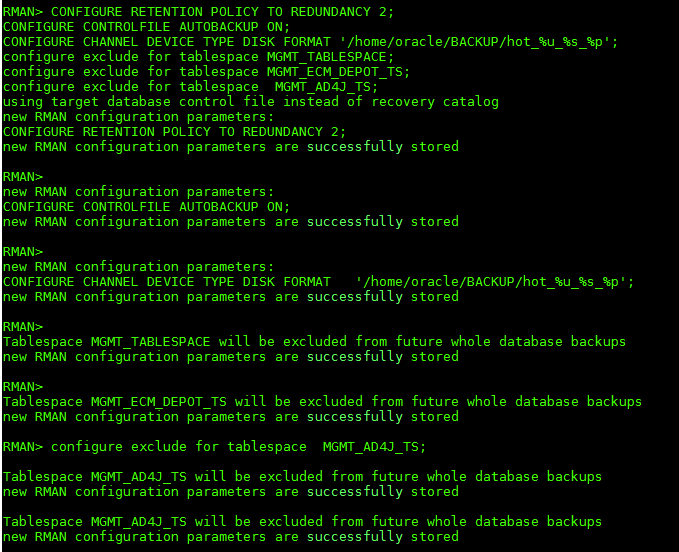
**You should create folders BACKUP and ARCHIVE and turn ON archiving here (if you did not do it so far), like shown in Session15.**

**Also, delete in RMAN all previous backup sets (if you have them) by using syntax DELETE BACKUPSET BS# (like shown in Session15). Then you may start this part.**

**a) Configure following options in RMAN**

* **Number of copies before becoming obsolete is 2**
* **Turn on auto backup of Control File**
* **Default folder to hold backup sets is /home/oracle/BACKUP and default name for these sets is hot\_%u\_%s\_%p**
* **Exclude all MGMT tablespaces from every backup**

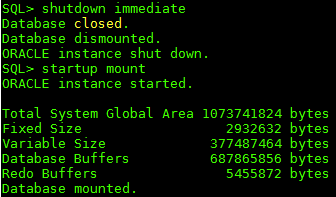
**CONFIGURE RETENTION POLICY TO REDUNDANCY 2; CONFIGURE CONTROLFILE AUTOBACKUP ON; CONFIGURE CHANNEL DEVICE TYPE DISK FORMAT '/home/oracle/BACKUP/hot\_%u\_%s\_%p'; configure exclude for tablespace MGMT\_TABLESPACE; configure exclude for tablespace MGMT\_ECM\_DEPOT\_TS; configure exclude for tablespace MGMT\_AD4J\_TS;**

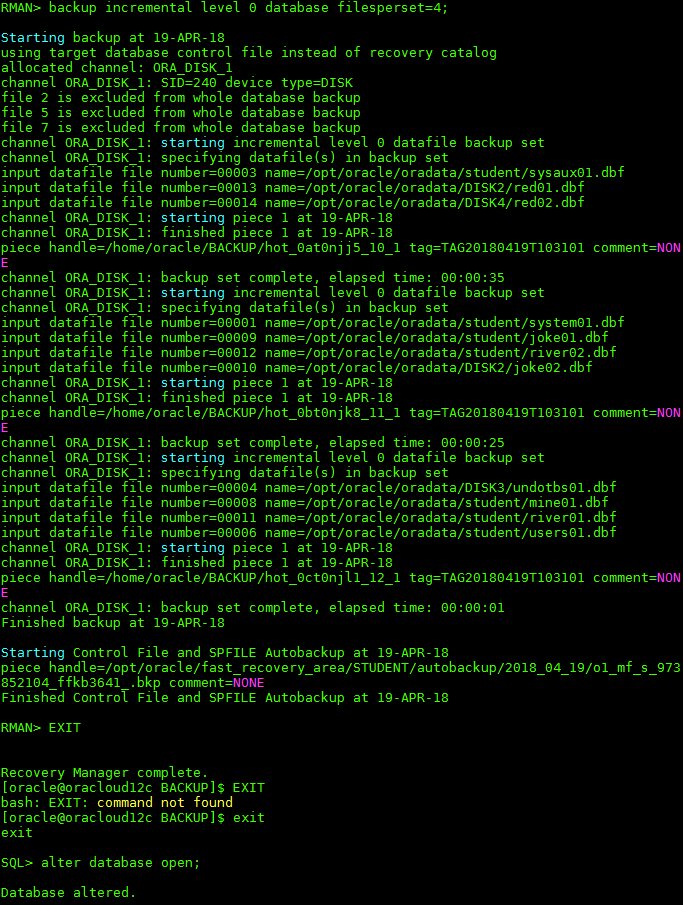


**b) Perform WHOLE, FULL and COLD database backup. This one may be used later as a base for Incremental backups. Allow only 4 files per set (peace)**

**Now show all backup sets created so far**

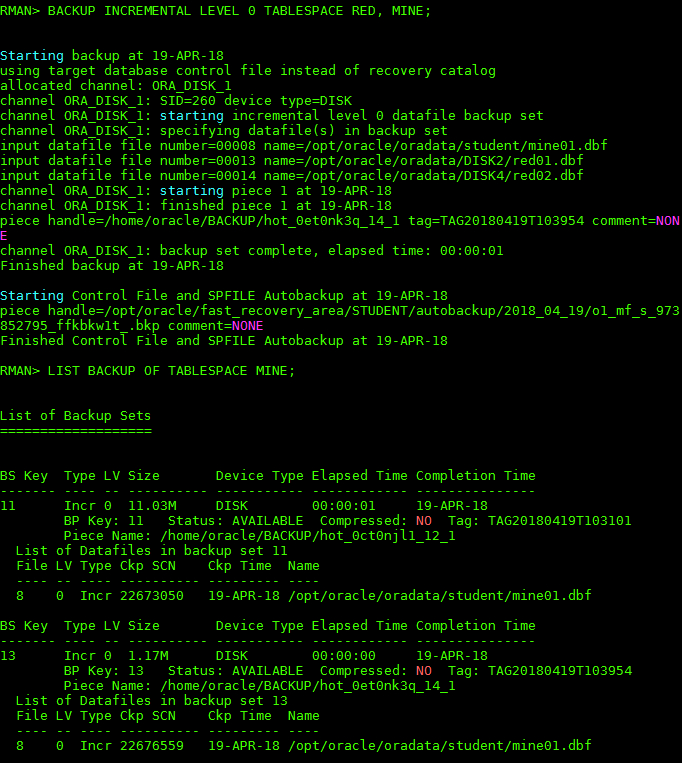
**shutdown immediate startup mount HOST rman target / backup incremental level 0 database filesperset=4; EXIT exit alter database open;**





**c) Perform HOT and Incremental-Cumulative backup of tablespaces MINE and RED Then show all backups of tablespace MINE**

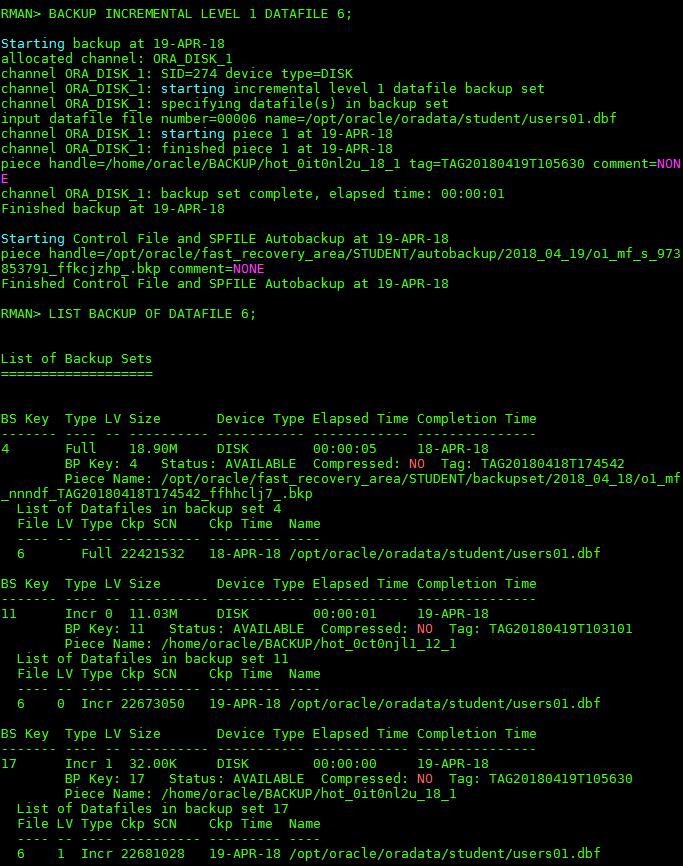
**BACKUP INCREMENTAL LEVEL 0 TABLESPACE RED, MINE; LIST BACKUP OF TABLESPACE MINE;**



**d) Perform HOT and Incremental-Differential backup of datafile that belongs to tablespace USERS. Then show all backups of that datafile**

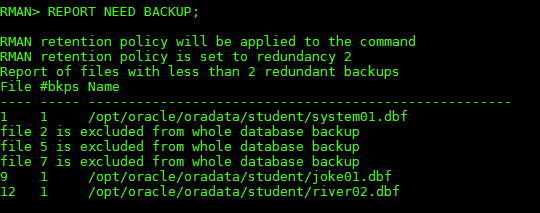
**SELECT FILE#, Name FROM v$datafile; BACKUP INCREMENTAL LEVEL 1 DATAFILE 6; LIST BACKUP OF DATAFILE 6;**





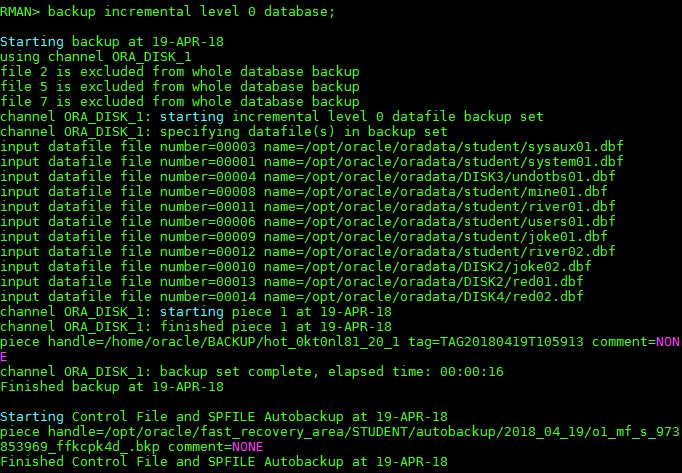
**e) Display all files that are due for backup regarding retention policy established in a)**

**REPORT NEED BACKUP;**



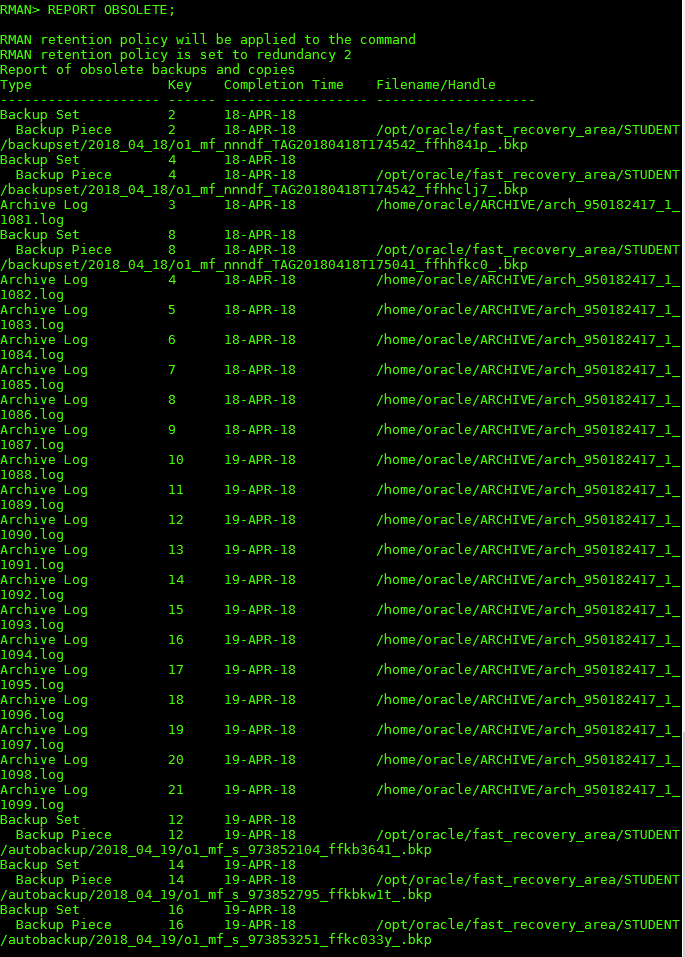
**f) Perform HOT and Incremental-Cumulative backup of database**

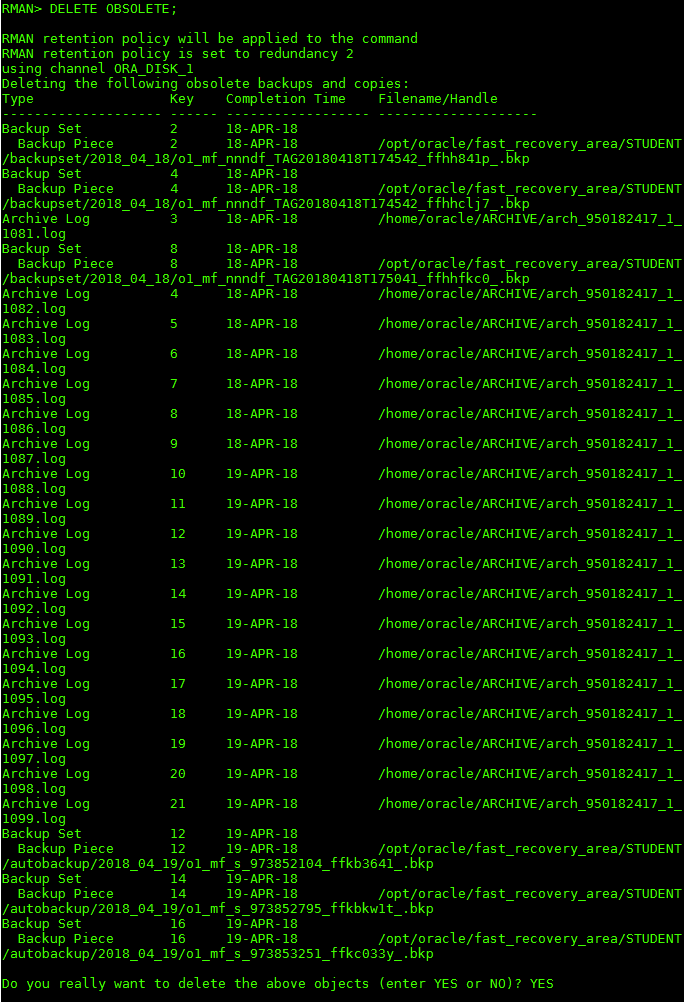
**backup incremental level 0 database;**



**g) Display all backup sets that are beyond retention policy established in a). Then delete all these sets.**

**REPORT OBSOLETE; DELETE OBSOLETE;**





**h) Now show all backup sets remained. Also, show the content of /home/oracle/BACKUP folder**

**list backup**

**HOST**

**ls -al**

