**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_expdp명령어 정리\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**--(리눅스에서 실행)**

$ expdp

userid = 유저명/비밀번호

compression = all | data\_only | metadata\_only | none

**--(어떤데이터를 압축할지 결정하는 옵션)**

cotent = all | data\_only | metadata\_only

**--(어떤데이터를 export 할지 정합니다.)**

directory = directory 명

**--(저장시킬 디렉토리명)**

version= 버전

**-- (말그대로 원하는 버전을 입력한다. 버전마다 데이터 구조가 다르다.)**

dumpfile = example.dmp

**-- (파일명 입력, [directory:]filename 도 가능하다.)**

encryption = all | data\_only | encrypted\_columns\_only | metadata\_only | none

**-- (내용을 덤프파일에 내려쓰기전 암호화 여부 옵션)**

encryption\_algorithm = aes128 | aes192c | aes256

**-- (암호화 알고리즘 옵션)**

encryption\_mode = dual | password | transparent

**--(암호화 복호화시 보안 타입을 명시**

**dual:import할때 wallet,password가능 ,**

**password : 비밀번호를 지정해 import할때 입력해야함**

**transparent : wallet을 제공할 경우 DBA의 중재없이 암호화된 덤프를 만들기 가능)**

encryption\_password = 비밀번호

**--(비밀번호 지정)**

estimate = block | statistics

**--(디스크사용량을 블록수 \* 블록사이즈로 표기할 것인지 , 사용량을 통계로 표현할 것인지 결정)**

estimate\_only = yes | no

**--(이 값을 yes로 설정하면 실제 export 작업은 하지 않고 export 작업이 진행될 경우 예상되는**

**디스크 사용량만 출력됩니다.)**

exclude = object\_type[:테이블명]

**--(export 작업 시 제외할 오브젝트를 선택합니다. )**

**--exclude=table:”\=’EMP’”  
--exclude=table:\”in \(\’emp\’,\’dept\’\)\”**

filesize = | B | KB | MB | GB | TB

**--(각각의 덤프파일의 최대용량을 지정합니다.)**flashback\_time = "timestamp 데이터"

**--(시간값에서 커밋이 일어난 가장 가까운 시간대의 데이터들을 export 합니다.)**

flashback\_scn = scn 값

**--(SCN으로 flashback을 작동시켜 과거시점의 데이터를 export합니다.)**

full = no | yes

**--(데이터베이스 전체를 export할 지 여부를 정합니다.)**

include = object\_type[:테이블명], ...]

**--(작업 시 해당 오브젝트를 포함시켜 작업합니다.)**

job\_name = sys\_export <mode> nn | 작업명

**--(작업명 지정)**

logfile = export.log

**--(로그파일의 파일명을 정합니다. 작업 도중 나는 에러등, 작업에 관련된 내용이 저장됩니다.)**

network\_link = DB link 명

**--(주어진 링크로 원격 DB에 작업을 합니다.)**

nologfile = yes | no

**--(log파일을 만들것인지 선택합니다.)**

parallel = 1 | n

**--(병렬처리 프로세스의 수 지정)**

parfile = 경로 +par파일명

**--( 작업을 수행할때 필요한 설정값을 파일에 저장하고 parfile 옵션값으로**

**파일을 지정하면 파일에 저장된 설정값을 지정할 수 있습니다.)**

query = where문

**--(조건을 줄수 있습니다.)**

**--(예) query = emp:\"where sal > 1000\" emp테이블에서 sal값이 1000보다 높은데이터만 export )**

reuse\_dumpfiles = yes | no

**--(같은명의 덤프파일이 있을경우 이값이 no 이면 에러, yes이면 덮어씁니다.)**

sample = [[스키마.]테이블:]n

**--(export할 데이터 전체 중 원하는 만큼만 작업합니다. 단위는 %{퍼센트})**

schemas = 현재유저 | 유저명

**--(지정한 스키마의 모든 오브젝트를 export합니다.)**

tables = [스키마.]테이블명 [:파티션명 ]

**--(지정한 테이블만 export합니다.)**

tablespaces = 테이블 스페이스명

**--(지정한 테이블 스페이스들을 export 합니다.)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_impdp명령어 정리\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

--(리눅스에서 실행 )

$ impdp

userid=유저명/비밀번호

content = all | data\_only | metadata\_only

**--(import할 데이터를 선택합니다. )**

data\_options = disable\_append\_hint | skip\_constraint\_errors

**--(import시 특정 데이터 타입을 제어할 지를 지정하는 옵션)**

directory =  data\_pump\_dir | DIRECTORY 오브젝트 이름

**--(디렉토리를 선택하는 옵션)**

dumpfile = expdat.dmp | [directory 오브젝트:]파일명

**--(덤프파일명 지정)**

encryption\_password = 비밀번호

**--(암호화된 파일일 때 비밀번호입력)**

exclude = object\_type[:테이블명], ...

**--(import 작업에서 제외시킬 데이터 지정)**

**--exclude=table:”\=’EMP’”  
--exclude=table:\”in \(\’emp\’,\’dept\’\)\”**

Flashback\_time = "timestamp 데이터"

**--(주어진 시간에 가까운 scn값으로 오브젝트를 과거시점으로 돌려 import)**

flashback\_scn = scn값

**--( scn값으로 오브젝트를 과거시점으로 돌려 import)**

full = no | yes

**--( 전체 데이터 베이스를 import 할 경우 yes)**

include = object\_type[:테이블명], ...

**--( 특정 대상을 import에 포함시킨다.)**

job\_name = sys\_export\_<mode>\_nn | 작업명

**--( 해당 import 작업에 이름 부여)**

logfile = export.log | 로그파일명

**--( log 파일지정 옵션)**

network\_link = DB Link명

**--( 기존 데이터베이스로부터 데이터베이스 링크를 통해 데이터 import 작업을 할 수 있도록 합니다.)**

nologfile = yes | no

**--( yes값을 입력하면 logfile이 생성되지 않음)**

parallel = 1 | n

**--( 병렬처리를 위한 프로세스 개수 지정)**

parfile = 경로+par파일명

**--( 옵션값을 파일에 저장해 이 옵션으로 그 파일을 지정하면 옵션적용가능 )**

partition\_options = none | departition | merge

**--( import중 어떻게 테이블 파티션들을 만들지 정한다.)**

query = where문

**--( import 시 테이블에 있는 데이터들을 where문으로 필터링한다/.)**

remap\_data = [스키마.]테이블.칼럼명:[스키마.]패키지.함수

**--( import되는 데이터들을 함수를 이용해 새로운 값으로 import가능)**

remap\_datafile = 기존 데이터파일:새로운 데이터파일

**--( 데이터를 저장할 데이터파일명을 새로운 데이터파일로 지정)**

remap\_schema = 기존 스키마:새로운 스키마

**--( 새로운 스키마 지정)**

remap\_table = [스키마.]기존 테이블명[.파티션명]:새로운 테이블명

**--( 새로운 테이블 지정)**

remap\_tablespace = 기존 테이블스페이스명:새로운 테이블스페이스명

**--(변경할 테이블 스페이스 지정)**

schemas = 현재유저​ | 유저명 [, ...]

**--( import할 스키마 지정)**

skip\_unusable\_indexes = yes | no

**--( 인덱스의 상태가 unusabled 인 인덱스들을 import에 포함시키지 않는다.)**

sqlfile = [DIRECTORY 오브젝트:]파일명

**--(DDL 문장을 따로 저장할 파일을 지정합니다.)**

status = 0 | n

**--(n초마다 import JOB의 상태를 보여준다.)**

table\_exists\_action = skip | append | truncate | replace

**--(동일한 테이블이 있을시 해당조치를 취할 수 있음)**

**--(replace는 metadata도 같이 적용해야함)**

tables = [스키마.]테이블명[:파티션명] [, ...]

**--(import할 테이블 지정)**

tablespaces = 테이블스페이스명[, ...]

**--(import할 테이블스페이스 지정)**

transport\_tablespaces = 테이블스페이스명 [, ...]

**--(DB LINK를 통하여 데이터를 import 작업을 할 테이블 스페이스명을 지정합니다.)**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_실습 명령어\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SELECT \* FROM dba\_directories;

**-- expdp 명령어를 실습할 scott유저 생성**

CREATE USER SCOTT

IDENTIFIED BY tiger

DEFAULT TABLESPACE users;

select \* from dba\_tablespaces ;

**-- 로그인, 리소스 권한 부여**

grant connect, resource to scott;

select \* from user\_tables;

**-- 디폴트 테이블 스페이스 확인(디폴트 테이블 스페이스가 다르면 오류 발생)**

select

username

,default\_tablespace

,temporary\_tablespace

from dba\_users

where username = 'SCOTT' or username = 'HR';

**-- 디폴트 설정**

alter user HR default tablespace USERS;

**-- 디렉토리를 만든다.**

**--디렉토리: DB에서 물리적 저장곤간을 접근할 수 있게 지정함**

**--expdp등에 사용됨**

create directory pump\_dir as '/app/oracle/19c/datapump';

**-- 다시 실습할때를위한 삭제 명령**

drop directory pump\_dir;

**-- 리눅스 커맨드라인에서 입력**

**-- 디렉토리에 hr.dmp이란 이름의 파일로 테이블 hr.emp,hr.dept 저장**

expdp HR/HR dumpfile = hr.dmp directory = pump\_dir tables=hr.emp,hr.dept;

**-- 리눅스 커맨드라인에서 입력**

**-- 디렉토리에 test1.dmp이란 이름의 파일로 테이블 test1.DDL\_SCRIPTS 저장**

expdp test1/oracle dumpfile = test1.dmp directory = pump\_dir tables=test1.DDL\_SCRIPTS;

**-- 테이블 스페이스 크기제한을 무한으로 늘린다.**

alter user scott default tablespace USERS quota unlimited on users;

**-- 디렉토리 읽는 권한을 준다.**

grant read, write on directory pump\_dir to scott;

**--**

**-- EXP\_FULL\_DATABASE 데이터베이스 익스포트 권한**

**-- IMP\_FULL\_DATABASE 데이터베이스 임포트 권한**

**-- 리눅스 커맨드라인에서 입력**

**-- import scott 계정에 hr.dmp 파일의 table hr.emp,hr.dept 정보 업로드**

impdp scott/tiger dumpfile = hr.dmp directory = pump\_dir tables=hr.emp,hr.dept ;

**-- content=data\_only 오직 데이터만 업로드**

impdp scott/tiger dumpfile = hr.dmp directory = pump\_dir tables=hr.emp,hr.dept content=data\_only;

**-- remap\_schema = hr:scott hr 스키마의 테이블을 scott 스키마로 임포트**

impdp scott/tiger dumpfile = hr.dmp directory = pump\_dir tables=hr.emp,hr.dept remap\_schema = hr:scott;

drop user scott;

[오라클 데이터 마이그레이션 (1부) : 네이버 블로그 (naver.com)](https://m.blog.naver.com/musicovery12/220248763142)