1. **Only 3 option come with a column line: formart heading and null**
2. **Skip n : the number of lines to skip**

**Original File:** [FINAL\_REVIEW.docx](https://cms2.webstudy.com/wsserveBinaryFile/servebinaryfile.ashx?type=M&userID=594590&courseID=81681&materialID=847092&hash=cc33e7bba90e9159adc26392cd3cefed)  
 **Size:** 94.7 KB

CREATE A TABLE

CODE 1

CREATE TABLE cdcases specifically usual

(caseid number(5,0),

 casename varchar2(30),

 casebought date);

DESCRIBE cdcases

CODE 1B

SELECT \* FROM user\_tables;

DROP TABLE cdcases purge;

describe cdcases

NOW RECREATE THE SAME TABLE

CREATE TABLE cdcases

(caseid number(5,0),

 casename varchar2(30),

 casedatebought date

);

CREATE TABLE albums

(albumid number(5,0),

 albumname varchar2(30),

 albumyear number(4,0),

 albumcost number(5,2)

);

CREATE TABLE cdcasesalbums

(caseid number(5,0),

 albumid number(5,0)

);

CREATE TABLE songs

(songid number(5,0),

 songname varchar2(30),

 albumid number(5,0)

);

CODE 1C

CREATE TABLE cust\_mkt

AS (SELECT customer#, city, state, zip, referred

        FROM customers);

CREATE A FEW CONSTRAINTS ON THE TABLE

alter table cdcases

add constraint cdcases\_caseid\_pk PRIMARY KEY (caseid);

alter table albums

add constraint albums\_albumid\_pk PRIMARY KEY (albumid);

alter table cdcasesalbums

add constraint cdcasesalbums\_caseidalbumid\_pk PRIMARY KEY (caseid,albumid);

alter table songs

add constraint songs\_songid\_pk PRIMARY KEY (songid);

insert into cdcases(caseid, casename)

values (1,'Rock Albums');

1. Alter table Account

* Add constraint Account\_AccountID\_pk primary key(AccountID);

1. Alter table Account

* add constraint Account\_AccountNumber\_uk unique (AccountNumber);

1. alter table Account

* modify (AccountOwner not null);

1. alter table Account

* add constraint Account\_AccountValue\_ck
* check(AccountValue>=0);

Code 1

alter table cdcases

add constraint cdcases\_caseid\_pk PRIMARY KEY (caseid);

alter table albums

add constraint albums\_albumid\_pk PRIMARY KEY (albumid);

alter table cdcasesalbums

add constraint cdcasesalbums\_caseidalbumid\_pk PRIMARY KEY (caseid,albumid);

alter table songs

add constraint songs\_songid\_pk PRIMARY KEY (songid);

insert into cdcases(caseid, casename)

values (1,'Rock Albums');

Example 1

cdcases

caseidcasename

1Rock Albums

2 Jazz Albums

cdcasesalbums

caseidalbumid

11

21

I can not delete a row from cdcases if there are rows in the cdcasesalbums referencing it.

Code 2

alter table cdcases

add constraint cdcases\_casename\_uk UNIQUE (casename);

insert into cdcases

values (2,'Rock Albums');

Code 3

alter table cdcases

add constraint cdcases\_casename\_ck

CHECK (casename <> 'Country');

alter table songs

add constraint songs\_albumid\_ck

CHECK (albumid > 0

);

Code 4

alter table cdcases

modify (casename NOT NULL);

alter table albums modify (albumname NOT NULL);

alter table songs modify (songname NOT NULL);

Example 1

cdcases

caseidcasename

1Rock Albums

2 Jazz Albums

cdcasesalbums

caseidalbumid

11

21

I can not delete a row from cdcases if there are rows in the cdcasesalbums referencing it.

Code 2

alter table cdcases

add constraint cdcases\_casename\_uk UNIQUE (casename);

insert into cdcases

values (2,'Rock Albums');

Code 3

alter table cdcases

add constraint cdcases\_casename\_ck

CHECK (casename <> 'Country');

alter table songs

add constraint songs\_albumid\_ck

CHECK (albumid > 0);

Code 4

alter table cdcases

modify (casename NOT NULL);

alter table albums modify (albumname NOT NULL);

alter table songs modify (songname NOT NULL);

CREATE A SYNONYM FOR THE TABLE

Week 10

1. create synonym myrecipes

for recipes;

1. create public synonym ourrecipes

for recipes;

1. select recipeid, recipename, recipedate from myrecipes;
2. select recipeid, recipename, recipedate from ourrecipes;
3. drop synonym myrecipes;
4. drop public synonym ourRecipes;

CREATE AN INDEX FOR THE TABLE

CREATE SEQUENCE FOR A COLUMN

create sequence footballplayerid\_seq

increment by 1

start with 1

nocache

nocycle;

create sequence teamid\_seq

increment by 1

start with 1

no cache

no cycle;

insert into footballplayers(playerid, playerdesc, teamid)

values (footballplayerid\_seq.nextval, 'Eli Manning',teamid\_seq.nextval);

insert into footballplayers(playerid, playerdesc, teamid)

values (footballplayerid\_seq.nextval, 'Tikki Barber',teamid\_seq.currval);

alter sequence footballplayerid\_seq

increment by 5;

insert into footballplayers(playerid, playerdesc, teamid)

values (footballplayerid\_seq.nextval, 'Tony Romo',3);

drop sequence footballplayerid\_seq;

INDEXES

CREATE INDEX footballplayers\_playerdesc\_idx ON footballplayers(playerdesc);

CREATE BITMAP INDEX footballplayers\_teamid\_idx ON footballplayers(teamid);

SYNONYMS

CREATE SYNONYM players FOR footballplayers;

DROP SYNONYM players;

CREATE PUBLIC SYNONYM players FOR footballplayers;

DROP PUBLIC SYNONYM players;

CREATE VIEW FOR THE TABLE

PERFORM AN INSERT STATEMENT

insert into visitors

(VisitorID, VisitorName, VisitorIP, VisitorDateTime, VisitorComment)

values(224, 'lamm', 'legi', '10-Feb-2014', 'first comments');

insert into Visitors

(VisitorID, VisitorName, Visitorip, VisitorDateTime, VisitorComment)

values(224, 'Bob Smith', 'Sip', '08-feb-2013', ‘This is the second time’);

insert into Visitors

(VisitorID, VisitorName, Visitorip, VisitorDateTime, VisitorComment)

values(533, 'Stacy Mark', 'ALLy', '23-May-2012', ' This is the third time’);

insert into Visitors

(VisitorID, VisitorName, Visitorip, VisitorDateTime)

values(865, 'Sharry SALL', 'Mall', '24-Jun-2002');

insert into Visitors

(VisitorID, VisitorName, Visitorip, VisitorDateTime)

values(893, 'Verty LOO', 'apple', '19-dec-2004');

create table Visitors2

as(select VisitorID, VisitorName, VisitorIP, VisitorDateTime, VisitorComment

from Visitors);

PERFORM AN UPDATE STATEMENT

update Visitors

set VisitorName='OneVisitor'

where VisitorID='654';

update Visitors

set visitorComment='No Comment';

PERFORM A DELETE STATEMENT

 delete from Visitors2

where VisitorComment is null;

savepoint righthere;

delete from Visitors2;

rollback to righthere;

select VisitorName from Visitors;

WRITE SQL STATEMENTS TO DO THE FOLLOWING:

1. RETRIEVE DATA FROM A SINGLE TABLE

select title, producedby, movieyear from movies

where movieyear in (1981, 1988)

and price <10;

select title, producedby, movieyear from movies

where movieyear in (1981, 1988)

and price <10;

select title, producedby, movieyear from movies

where producedby ='Dennis Quaid'

or (producedby='Bill Murray' and movieyear=1981);

select title, producedby, movieyear from movies

where producedby like 'D%'

or movieyear=1996;

select title, producedby, movieyear from movies

where price >9

and title between 'B' and 'G';

select title, producedby, movieyear from movies

where producedby like 'R%';

select title, movieyear from movies

where movieyear like 'K%';

select title, movieyear from movies

where movieyear in (1981, 1986);

select title, movieyear from movies

where title between 'E' and 'H';

select title, movieyear from movies

where movieyear not between 1981 and 1989;

select title, movieyear from movies

where producedby !='bill murray';

select title, movierating from movies

where movierating is not null;

select title, movierating from movies

where movierating is null;

select title, movieyear from movies

where producedby<>'bill murray';

select title, movieyear from movies

where producedby !='bill murray';

select title, price from movies

order by price;

select title, price from movies

order by title desc;

select title, price, producedby from movies

order by producedby desc, title asc;

select title from movies

where statemade='PA'

or price>20;

select title, movieyear, producedby from movies

where statemade ='NJ'

or statemade ='PA'

order by 2 asc, 3 desc;

1. RETRIEVE DATA BY JOINING 2 TABLES TOGETHER

1

select employees.firstname, employees.lastname

from employees, emergencyinfo;

2

select employees.firstname, employees.lastname

from employees cross join emergencyinfo;

3

select firstname, lastname, deptname

from employees, departments

where employees.deptid=departments.departmentid;

4

select deptname, firstname, lastname

from departments, employees

where employees.deptid=departments.departmentid

and deptname='Development';

5

select deptname, firstname, lastname

from departments natural join employees;

select \* from artists;

select \* from albums;

-- Pure SQL way to to a Cartesian example

select artistname, title

from albums, artists;

-- Joins both tables rows to every other row example

select artistname, title

from artists cross join albums

order by artistname, title;

-- Example of inner join

select artistname, title

from artists, albums

where artists.artistid = albums.artistid;

-- Easier to do with Alias's

select artistname, title

from artists ar, albums al

where ar.artistid = al.artistid;

-- Ambiguous Name Example

select artistid, artistname, title

from artists, albums

where artists.artistid = albums.artistid;

-- How to fix Ambiguous Field

select ar.artistid, ar.artistname, al.title

from artists ar, albums al

where ar.artistid = al.artistid;

-- Additional conditions

select ar.artistid, ar.artistname, al.title

from artists ar, albums al

where ar.artistid = al.artistid;

and al.artistid = 1;

-- Joining 3 tables

select ar.artistname, al.title, s.title

from artists ar, albums al, songs s

where ar.artistid = al.artistid

and al.albumid = s.albumid

order by 1,2,3;

-- Natural join example

select artistname, title

from albums natural join artists;

-- Using keyword --> artistid exists in both tables

select artistname, title

from albums join artists using (artistid);

-- On keyword, why is it needed? -> albumid and albid

-- reference the same column, not the same name though

select al.title, s.title

from albums al join songs s on al.albumid = s.albid;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- Self Select

select a1.artistname, a2.artistname

from artists a1, artists a2

where a1.artiststartedbyid = a2.artistid;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

select a1.artistname, a2.artistname

from artists a1, artists a2

where a1.artiststartedbyid = a2.artistid (+);

select a1.artistname, a2.artistname

from artists a1 left outer join artists a2

on a1.artiststartedbyid = a2.artistid;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

select artistname

from artists

where artistid = 1

union

select artistname

from artists;

select artistname

from artists

where artistid = 1

union all

select artistname

from artists;

select artistname

from artists

where artistid = 1

intersect

select artistname

from artists;

select artistname

from artists

minus

select artistname

from artists

where artistid = 1;

select albums.title, songs.title

from albums join songs ON albums.albumid=songs.albid;

Join 3 tables

Referring

select a1.artistname, a2.artistname from artists a1, artists a2

where a1.artiststartedbyid=a2.artistid;

select a1.artistname, a2.artistname from artists a1, artists a2

where a1.artiststartedbyid=a2.artistid (+);

select a1.artistname, a2.artistname

from artists a1 left outer join artists a2

on a1.artiststartedbyid = a2.artistid;

select artistname from artists

where artistid=2

union

select artistname from artists;

select artistname from artists

where artistid=2

union all

select artistname from artists;

Write a select statement to return the name of the album and song where the song title begins with an N with just a where clause.

select albums.title, songs.title

from albums, songs

where albums.albumid=songs.albid;

select albums.title, songs.title

from albums, songs

where albums.albumid=songs.albid

and songs.title like 'N%';

Write the same select statement using a JOIN and ON statement

select albums.title, songs.title

from albums join songs

on albums.albumid=songs.albid

and songs.title like 'N%';

Write a sql statement that returns a list of all songs that begin with an ‘N’ and another sql statement that begins with a ‘T’. You should combine the results of both statements and return the results in ascending order. There should be no duplicate rows returned.

select title from songs

where title like 'N%'

union

select title from songs

where title like 'T%';

select title from songs

where title like 'N%'

union

select title from songs

where title like 'T%'

order by title asc;

3.   RETRIEVE DATA UTILIZING SINGLE ROW FUNCTIONS

4.   RETRIEVE DATA AND GROUP THE DATA

5.   WRITE A STATEMENT USING A SUBQUERY

Week 7

* Create user pierreking Identify by password;
* GRANT CREATE SESSION To pierreking

What does DML Stand For

How do you create a user in Oracle?

Cartesian Join, Natural Join, Left Outer Join, Output to a file