Database Programming

Ensuring Quality Query Results - Advanced Techniques

Objectives

This lesson covers the following objectives:

- Create an advanced query to produce specified data
- Modify an advanced query to produce specified data

Purpose

You've learned the syntax rules for generating a SQL query, but are you sure you are producing the desired data? Looking at the desired output and then figuring out the query to generate that output helps you to gain confidence that your query results are what you expect.

Create These Tables

```
create table emp as select * from employees;
create table dept as select * from departments;
```

Write the Query

Problem:

Produce a report that lists the constraint name, type, column name, and column position of all the constraints on the JOB_HISTORY table, apart from the not null constraints.

Tables Used:

user_constraints,

user_cons_columns

CONSTRAINT_NAME	CONSTRAINT_TYPE	COLUMN_NAME	POSITION
JHIST_EMP_ID_ST_DATE_PK	P	EMPLOYEE_ID	1
JHIST_EMP_ID_ST_DATE_PK	P	START_DATE	2
JHIST_JOB_FK	R	JOB_ID	1
JHIST_EMP_FK	R	EMPLOYEE_ID	1
JHIST_DEPT_FK	R	DEPARTMENT_ID	1

Create the Statement

Create a primary key constraint on the emp table's employee_id column.

```
Table altered.
```

Create the Statement

Create a primary key on the dept table's department_id column.

```
Table altered.
```

Fix the Code

Problem:

Add a foreign constraint between DEPT and EMP so that only valid departments can be entered in the EMP table, but make sure you can delete any row from the DEPT table.

Statement: ALTER TABLE e_mp

CREATE CONTRAINT FOREIGN KEY (dept_id) REFS dept(deptid) on del cascade

```
Table altered.
```

Create the Code

Test the foreign key constraint you just created by following the examples on this slide.

Examine the number of rows in the EMP table. Remove the details of department 10 from the dept table.

Now count emps again and check if there are fewer employees as well.

```
SELECT COUNT(*) "No of Emps"
FROM emp;
```

Results | Explain | Describe | Saved SQL | His

No of Emps
20

```
DELETE dept
WHERE  department_id = 10
```

Results | Explain | Describe | Saved SQL

```
1 row(s) deleted.
```

```
SELECT COUNT(*) "No of Emps"
FROM emp;
```

Results | Explain | Describe | Saved SQ

No of Emps
19

Write the Query

Problem:

Produce a report that returns the last name, salary, department number, and average salary of all the departments where salary is greater than the average salary.

LAST_NAME	SALARY	DEPARTMENT_ID	SALAVG
Hartstein	13000	20	9500
Mourgos	5800	50	3500
Hunold	9000	60	6400
Zlotkey	10500	80	10033
Abel	11000	80	10033
King	24000	90	19333
Higgins	12000	110	10150

Tables Used:

Employees, Departments

Write the Code

Problem:

Create a view named V2 that returns the highest salary, lowest salary, average salary, and department name.

Tables Used:

emp, dept

```
SELECT * from v2
```

Results | Explain | Describe | Saved SQL | History

Department Name	Lowest Salary	Highest Salary	Average Salary
Accounting	8300	12000	10150
IT	4200	9000	6400
Executive	17000	24000	19333
Shipping	2500	5800	3500
Sales	8600	11000	10033
Marketing	6000	13000	9500

Write the Code

Problem:

Create a view named Dept_Managers_view that returns a listing of department names along with the manager initial and surname for that department. Test the view by returning all the rows from it. Make sure no rows can be updated through the view.

Try to run an UPDATE statement against the view.

```
View created.
```

DEPT_NAME	MGR_NAME
Executive	S.King
IT	A.Hunold
Shipping	K.Mourgos
Sales	E.Zlotkey
Administration	J.Whalen
Marketing	M.Hartstein
Accounting	S.Higgins

Tables Used:

Employees, departments

```
ORA-01733: virtual column not allowed here
```

Fix the Code

Problem:

The following statement contains errors.

Fix them and run the code to get the displayed result.

Statement:

DROP V3 views;

```
View dropped.
```

Create a Sequence and Fix the Code

Problem:

Create a sequence named ct_seq with all the default values. Run the statements and fix the error. Correct the statement to return the subsequent number.

Code:

```
CREATE SEQUENCE ct_seq;

SELECT ct_seq.currval
FROM dual;
```

```
Sequence created.
```

ORA-08002: sequence CT_SEQ.CURRVAL is not yet defined in this session

Fix the Code

Problem:

Look at the insert statement and fix the error.

Code:

```
INSERT emp
(employee_id,first_name,last_name,email,phone_number,
hire_date,job_id,salary,commission_pct,manager_id,department_id)
VALUS
(currval.ct_seq,'Kaare’,'Hansen’,'KHANSEN’,'44965 832123’,sysdate
,'Manager’,6500,null,100,10)
```

```
ORA-00984: column not allowed here
```

Fix the Code

Problem:

Fix the error in the SQL statement to create the index as shown in the screenshot.

Code:

```
CREATE INX emp indx FOR TABLE emp(employee_id DESC,
UPPR(SUBST(firstname,1.1 ||” “||astname)
```

TABLE_NAME	INDEX_NAME	INDEX_TYPE	COLUMN_EXPRESSION	COLUMN_POSITION
EMP	EMP_INDX	FUNCTION-BASED NORMAL	"EMPLOYEE_ID"	1
EMP	EMP_INDX	FUNCTION-BASED NORMAL	UPPER(SUBSTR("FIRST_NAME",1,1)||' '||"LAST_NAME")	2

Write the Code

Problem:

Write the SQL statement to list all the user tables which contain the name PRIV.

Tables Used:

dictionary

TABLE_NAME	COMMENTS
USER_AQ_AGENT_PRIVS	-
USER_COL_PRIVS	Grants on columns for which the user is the owner, grantor or grantee
USER_COL_PRIVS_MADE	All grants on columns of objects owned by the user
USER_COL_PRIVS_RECD	Grants on columns for which the user is the grantee
USER_REPGROUP_PRIVILEGES	Information about users who are registered for object group privileges
USER_ROLE_PRIVS	Roles granted to current user
USER_RSRC_CONSUMER_GROUP_PRIVS	Switch privileges for consumer groups for the user
USER_RSRC_MANAGER_SYSTEM_PRIVS	system privileges for the resource manager for the user
USER_SYS_PRIVS	System privileges granted to current user
USER_TAB_PRIVS	Grants on objects for which the user is the owner, grantor or grantee
USER_TAB_PRIVS_MADE	All grants on objects owned by the user
USER_TAB_PRIVS_RECD	Grants on objects for which the user is the grantee

Fix the Code

Problem:

Give select access to public on the EMP table, and verify the grant by running this query. The query contains errors that you must fix before you can run the select statement.

Code:

```
GRANT SELECT ON emp TO PUBLIC

SELECT *
FROM   usr_tab_privs
WHERE tablename = “emp”
```

```
Statement processed.
```

GRANTEE	OWNER	TABLE_NAME	GRANTOR	PRIVILEGE	GRANTABLE	HIERARCHY
PUBLIC	US_CURR1_SQL01_T01	EMP	US_CURR1_SQL01_T01	SELECT	NO	NO

Write the Code

Problem:

Complete the following query using regular expressions to return only the numbers from the following string: 'Oracle Academy9547d6905%&^ db apex'.

Statement:

```
SELECT REGEXP_REPLACE('Oracle Academy9547d6905%&^ db apex' YOUR CODE HERE)
regexpreplace
FROM DUAL
```

REGEXPREPLACE
95476905

Write the Code

Problem:

Amend the previous query using regular expressions to return the number of digits from the following string: 'Oracle Academy9547d6905%&^ db'

Statement:

```
SELECT REGEXP_REPLACE('Oracle
Academy9547d6905%&^ db apex' YOUR CODE HERE)
regexpreplace
FROM DUAL
```

REGEXPREPLACE
8

Write the Code

Problem:

Amend the query again to return only the non-numeric characters.

Statement:

```
SELECT REGEXP_REPLACE('Oracle Academy9547d6905%&^ db apex' YOUR CODE HERE)
regexpreplace
FROM DUAL
```

REGEXPREPLACE
Oracle Academyd%&^ db apex

Write the Code

Problem:

Using Oracle proprietary joins, construct a statement that returns all the employee_id's joined to all the department_names.

Tables Used:

Employees, departments

202	Accounting
205	Accounting
206	Accounting
100	Contracting
101	Contracting
102	Contracting
103	Contracting
104	Contracting
107	Contracting
124	Contracting
141	Contracting
142	Contracting
143	Contracting
144	Contracting
149	Contracting
174	Contracting
176	Contracting
178	Contracting
200	Contracting
201	Contracting
202	Contracting
205	Contracting
206	Contracting

160 rows returned in 0.01 seconds

Write the Code

Problem:

Still using Oracle Joins, correct the previous statement so that it returns only the name of the department that the employee actually works in.

Tables Used:

Employees, departments

EMPLOYEE_ID	DEPARTMENT_NAME
200	Administration
201	Marketing
202	Marketing
124	Shipping
144	Shipping
143	Shipping
142	Shipping
141	Shipping
107	IT
104	IT
103	IT
174	Sales
149	Sales
176	Sales
102	Executive
100	Executive
101	Executive
205	Accounting
206	Accounting

19 rows returned in 0.03 seconds

Write the Code

Problem:

Still using Oracle Joins, construct a query that lists the employees last name, the department name, the salary and the country name of all employees.

Tables Used:

Employees, departments, locations and countries

LAST_NAME	DEPARTMENT_NAME	SALARY	COUNTRY_NAME
King	Executive	24000	United States of America
Kochhar	Executive	17000	United States of America
De Haan	Executive	17000	United States of America
Whalen	Administration	4400	United States of America
Higgins	Accounting	12000	United States of America
Gietz	Accounting	8300	United States of America
Zlotkey	Sales	10500	United Kingdom
Abel	Sales	11000	United Kingdom
Taylor	Sales	8600	United Kingdom
Mourgos	Shipping	5800	United States of America
Rajs	Shipping	3500	United States of America
Davies	Shipping	3100	United States of America
Matos	Shipping	2600	United States of America
Vargas	Shipping	2500	United States of America
Hunold	IT	9000	United States of America
Ernst	IT	6000	United States of America
Lorentz	IT	4200	United States of America
Hartstein	Marketing	13000	Canada
Fay	Marketing	6000	Canada

19 rows returned in 0.76 seconds Download

Write the Code

Problem:

Still using Oracle join syntax, alter the previous query so that it also includes the employee record of the employee with no department_id, 'Grant'.

Tables Used:

Employees, departments, locations and countries

LAST_NAME	DEPARTMENT_NAME	SALARY	COUNTRY_NAME
Hartstein	Marketing	13000	Canada
Fay	Marketing	6000	Canada
Zlotkey	Sales	10500	United Kingdom
Abel	Sales	11000	United Kingdom
Taylor	Sales	8600	United Kingdom
Hunold	IT	9000	United States of America
Ernst	IT	6000	United States of America
Lorentz	IT	4200	United States of America
Mourgos	Shipping	5800	United States of America
Rajs	Shipping	3500	United States of America
Davies	Shipping	3100	United States of America
Matos	Shipping	2600	United States of America
Vargas	Shipping	2500	United States of America
Higgins	Accounting	12000	United States of America
Gietz	Accounting	8300	United States of America
King	Executive	24000	United States of America
Kochhar	Executive	17000	United States of America
De Haan	Executive	17000	United States of America
Whalen	Administration	4400	United States of America
Grant	-	7000	-

20 rows returned in 0.02 seconds Download

Summary

In this lesson, you should have learned how to:

- Create an advanced query to produce specified data
- Modify an advanced query to produce specified data