**LAB 3.1**

SQL> SELECT \* FROM HR.DEPARTMENTS;

PRESS RETURN TO CONTINUE

DEPARTMENT\_ID DEPARTMENT\_NAME MANAGER\_ID LOCATION\_ID

------------- ------------------------------ ---------- -----------

10 Administration 200 1700

20 Marketing 201 1800

30 Purchasing 114 1700

40 Human Resources 203 2400

50 Shipping 121 1500

60 IT 103 1400

70 Public Relations 204 2700

80 Sales 145 2500

90 Executive 100 1700

100 Finance 108 1700

110 Accounting 205 1700

120 Treasury 1700

130 Corporate Tax 1700

140 Control And Credit 1700

150 Shareholder Services 1700

160 Benefits 1700

170 Manufacturing 1700

180 Construction 1700

190 Contracting 1700

200 Operations 1700

210 IT Support 1700

220 NOC 1700

230 IT Helpdesk 1700

240 Government Sales 1700

250 Retail Sales 1700

260 Recruiting 1700

270 Payroll 1700

27 rows selected.

SQL> SPOOL OFF

**LAB 3.2**

SQL> SELECT COUNT(EMPLOYEE\_NAME)

2 FROM HR.EMPLOYEES

3 WHERE SALARY > 15000 AND HIRE\_DATE BETWEEN '01-JAN-02' AND '01-JAN-05'

4 ;

SELECT COUNT(EMPLOYEE\_NAME)

\*

ERROR at line 1:

ORA-00904: "EMPLOYEE\_NAME": invalid identifier

SQL> SELECT COUNT(EMPLOYEE\_ID)

2 FROM HR.EMPLOYEES

3 WHERE SALARY >15000 AND HIRE\_DATE BETWEEN

4 '01-JAN-02' AND '01-JAN-05';

PRESS RETURN TO CONTINUE

COUNT(EMPLOYEE\_ID)

------------------

1

SQL> SPOOL OFF

**Lab 3.3**

SQL> SELECT COUNT(PHONE\_NUMBER)

2 FROM HR.EMPLOYEES

3 WHERE PHONE\_NUMBER NOT LIKE '515.%%%.%%%%';

PRESS RETURN TO CONTINUE

COUNT(PHONE\_NUMBER)

-------------------

86

SQL> SPOOL OFF

**Lab 3.4**

SQL> SELECT FIRST\_NAME||LAST\_NAME

2 FROM HR.EMPLOYEES

3 WHERE DEPARTMENT\_ID = 100

4 ORDER BY FIRST\_NAME;

PRESS RETURN TO CONTINUE

FIRST\_NAME||LAST\_NAME

---------------------------------------------

DanielFaviet

IsmaelSciarra

JohnChen

Jose ManuelUrman

LuisPopp

NancyGreenberg

6 rows selected.

SQL> SPOOL OFF

**Lab 3.5**

SQL> SELECT CITY, STATE\_PROVINCE, COUNTRY\_NAME

2 FROM COUNTRIES

3 JOIN LOCATIONS USING (COUNTRY\_ID)

4 JOIN REGIONS USING (REGION\_ID)

5 WHERE REGION\_ID = 3;

PRESS RETURN TO CONTINUE

CITY STATE\_PROVINCE COUNTRY\_NAME

------------------------------ ------------------------- ----------------------------------------

Tokyo Tokyo Prefecture Japan

Hiroshima Japan

Beijing China

Bombay Maharashtra India

Sydney New South Wales Australia

Singapore Singapore

6 rows selected.

SQL> SPOOL OFF

**Lab 3.6**

SQL> SELECT \* FROM HR.LOCATIONS

2 WHERE STATE\_PROVINCE IS NULL;

PRESS RETURN TO CONTINUE

LOCATION\_ID STREET\_ADDRESS POSTAL\_CODE CITY STATE\_PROVINCE CO

----------- ---------------------------------------- ------------ ------------------------------ ------------------------- --

1000 1297 Via Cola di Rie 00989 Roma IT

1100 93091 Calle della Testa 10934 Venice IT

1300 9450 Kamiya-cho 6823 Hiroshima JP

2000 40-5-12 Laogianggen 190518 Beijing CN

2300 198 Clementi North 540198 Singapore SG

2400 8204 Arthur St London UK

6 rows selected.

SQL> SPOOL OFF

**Lab 3.7**

SQL> SELECT 2+2

2 FROM DUAL

3 ;

PRESS RETURN TO CONTINUE

2+2

----------

4

SQL> SELECT 1

2 FROM DUAL;

PRESS RETURN TO CONTINUE

1

----------

1

SQL> SELECT USER

2 FROM DUAL;

PRESS RETURN TO CONTINUE

USER

------------------------------

YJARRAR

SQL> SELECT SYSDATE

2 FROM DUAL;

PRESS RETURN TO CONTINUE

SYSDATE

---------

22-JAN-19

SQL> SELECT \* FROM

2 DUAL;

PRESS RETURN TO CONTINUE

D

-

X

SQL> SPOOL OFF

* The dual table was created as an object from within the Oracle Data Dictionary. It was never meant to be seen (according to Charles Weiss creator of SQL\*Plus). It was meant so that you could JOIN to the DUAL table and create 2 rows in the result of every one row in your table. Then by using the GROUP BY call; the results could join to show the amount of data and index extents. The table is consisted of a name called DUMMY and a value called ‘x’.