Introduction to Oracle9*i*: SQL

Additional Practices • Volume 3

40049GC10
Production 1.0
June 2001
D33053

ORACLE®

Authors

Nancy Greenberg
Priya Nathan

Technical Contributors and Reviewers

Josephine Turner
Martin Alvarez
Anna Atkinson
Don Bates
Marco Berbeek
Andrew Brannigan
Michael Gerlach
Sharon Gray
Rosita Hanoman
Mozhe Jalali
Sarah Jones
Charbel Khouri
Christopher Lawless
Diana Lorentz
Nina Minchen
Cuong Nguyen
Daphne Nougier
Patrick Odell
Laura Pezzini
Stacey Procter
Maribel Renau
Bryan Roberts
Helen Robertson
Sunshine Salmon
Casa Sharif
Bernard Soleillant
Craig Spoonemore
Ruediger Steffan
Karla Villasenor
Andree Wheeley
Lachlan Williams

Publisher

Sheryl Domingue

Contents

Preface

Curriculum Map

2 Restricting and Sorting Data

3 Single-Row Functions

Additional Practices

These exercises can be used for extra practice after you have discussed the following topics: basic SQL SELECT statement, basic *i*SQL*Plus commands, and SQL functions.

1. Show all data of the clerks who have been hired after the year 1997.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DAT
143	Randall	Matos	RMATOS	650.121.2874	15-MAR-96
144	Peter	Vargas	PVARGAS	650.121.2004	09-JUL-98

2. Show the last name, job, salary, and commission of those employees who earn commission. Sort the data by the salary in descending order.

LAST_NAME	JOB_ID	SALARY	COMMISSION_PCT
Abel	SA_REP	11000	.3
Zlotkey	SA_MAN	10500	.2
Taylor	SA_REP	8600	.2
Grant	SA_REP	7000	.15

3. Show the employees that have no commission with a 10% raise in their salary (round off the salaries).

New salary
The salary of King after a 10% raise is 26400
The salary of Kochhar after a 10% raise is 18700
The salary of De Haan after a 10% raise is 18700
The salary of Hunold after a 10% raise is 9900
The salary of Ernst after a 10% raise is 6600
The salary of Lorentz after a 10% raise is 4620
The salary of Mourgos after a 10% raise is 6380
The salary of Rajs after a 10% raise is 3850
The salary of Davies after a 10% raise is 3410
The salary of Matos after a 10% raise is 2860
The salary of Vargas after a 10% raise is 2750
The salary of Whalen after a 10% raise is 4840
The salary of Hartstein after a 10% raise is 14300
The salary of Fay after a 10% raise is 6600
New salary
The salary of Higgins after a 10% raise is 13200
The salary of Gietz after a 10% raise is 9130

16 rows selected.

4. Show the last names of all employees together with the number of years and the number of completed months that they have been employed.

LAST_NAME	YEARS	MONTHS
King	13	9
Kochhar	11	5
De Haan	8	2
Hunold	11	2
Ernst	9	9
Lorentz	2	1
Mourgos	1	4
Rajs	5	5
Davies	4	1
Matos	3	0
Gietz	6	9

20 rows selected.

5. Show those employees that have a name starting with *J*, *K*, *L*, or *M*.

LAST_NAME
King
Kochhar
Lorentz
Mourgos
Matos

6. Show all employees, and indicate with "Yes" or "No" whether they receive a commission.

LAST_NAME	SALARY	COM
King	24000	No
Kochhar	17000	No
De Haan	17000	No
Hunold	9000	No
Ernst	6000	No
Lorentz	4200	No
Mourgos	5800	No
Rajs	3500	No

(**Note:** results continued on next page)

Davies	3100	No
Matos	2600	No
Vargas	2500	No
Zlotkey	10500	Yes
Abel	11000	Yes
Taylor	8600	Yes

LAST_NAME	SALARY	COM
Grant	7000	Yes
Whalen	4400	No
Hartstein	13000	No
Fay	6000	No
Higgins	12000	No
Gietz	8300	No

20 rows selected.

These exercises can be used for extra practice after you have discussed the following topics: SQL basic SELECT statement, basic *i*SQL*Plus commands, SQL functions, joins, and group functions.

7. Show the department names, locations, names, job titles, and salaries of employees who work in location 1800.

DEPARTMENT_NAME	LOCATION_ID	LAST_NAME	JOB_ID	SALARY
Marketing	1800	Hartstein	MK_MAN	13000
Marketing	1800	Fay	MK_REP	6000

8. How many employees have a name that ends with an "*n*"? Create two possible solutions.

COUNT(*)
3

9. Show the names and locations for all departments, and the number of employees working in each department. Make sure that departments without employees are included as well.

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID	COUNT(E.EMPLOYEE_ID)
10	Administration	1700	1
20	Marketing	1800	2
50	Shipping	1500	5
60	IT	1400	3
80	Sales	2500	3
90	Executive	1700	3
110	Accounting	1700	2
190	Contracting	1700	0

8 rows selected.

10. Which jobs are found in departments 10 and 20?

JOB_ID
AD_ASST
MK_MAN
MK_REP

11. Which jobs are found in the Administration and Executive departments, and how many employees do these jobs? Show the job with the highest frequency first.

JOB_ID	FREQUENCY
AD_VP	2
AD_ASST	1
AD_PRES	1

These exercises can be used for extra practice after you have discussed the following topics: basic SQL SELECT statements, basic *i*SQL*Plus commands, SQL functions, joins, group functions, subqueries.

12. Show all employees who were hired in the first half of the month (before the 16th of the month).

LAST_NAME	HIRE_DATE
De Haan	13-JAN-93
Hunold	03-JAN-90
Lorentz	07-FEB-99
Matos	15-MAR-98
Vargas	09-JUL-98
Abel	11-MAY-96
Higgins	07-JUN-94
Gietz	07-JUN-94

8 rows selected.

13. Show the names, salaries, and the number of dollars (in thousands) that all employees earn.

LAST_NAME	SALARY	THOUSANDS
King	24000	24
Kochhar	17000	17
De Haan	17000	17
Hunold	9000	9
Ernst	6000	6
Lorentz	4200	4
Mourgos	5800	5

(**Note:** Results continue on the next page)

Rajs	3500	3
Davies	3100	3
Matos	2600	2
Vargas	2500	2
Zlotkey	10500	10
Abel	11000	11
Taylor	8600	8
LAST_NAME	SALARY	THOUSANDS
Grant	7000	7
Whalen	4400	4
Hartstein	13000	13
Fay	6000	6
Higgins	12000	12
Gietz	8300	8

20 rows selected.

13. Show all employees who have managers with a salary higher than $15,000. Show the following data: employee name, manager name, manager salary, and salary grade of the manager.

LAST_NAME	MANAGER	SALARY	GRA
Kochhar	King	24000	E
De Haan	King	24000	E
Mourgos	King	24000	E
Zlotkey	King	24000	E
Hartstein	King	24000	E
Whalen	Kochhar	17000	E
Higgins	Kochhar	17000	E
Hunold	De Haan	17000	E

8 rows selected.

14. Show the department number, name, number of employees, and average salary of all departments, together with the names, salaries, and jobs of the employees working in each department.

DEPARTMENT_ID	DEPARTMENT_NAME	EMPLOYEES	AVG_SAL	LAST_NAME	SALARY
10	Administration	1	4400.00	Whalen	4400
20	Marketing	2	9500.00	Fay	6000
				Hartstein	13000
50	Shipping	5	3500.00	Davies	3100
				Matos	2600
				Mourgos	5800
				Rajs	3500
				Vargas	2500
60	IT	3	6400.00	Ernst	6000
				Hunold	9000
				Lorentz	4200
80	Sales	3	10033.33	Abel	11000
				Taylor	8600
				Zlotkey	10500
DEPARTMENT_ID	DEPARTMENT_NAME	EMPLOYEES	AVG_SAL	LAST_NAME	SALARY
90	Executive	3	19333.33	De Haan	17000
				King	24000
				Kochhar	17000
110	Accounting	2	10150.00	Gietz	8300
				Higgins	12000
190	Contracting	0	No average		

20 rows selected.
breaks cleared

15. Show the department number and the lowest salary of the department with the highest average salary.

DEPARTMENT_ID	MIN(SALARY)
90	17000

16. Show the department numbers, names, and locations of the departments where no sales representatives work.

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
90	Executive	100	1700
110	Accounting	205	1700
190	Contracting		1700

7 rows selected.

17. Show the department number, department name, and the number of employees working in each department that:

    a. Includes fewer than 3 employees:

DEPARTMENT_ID	DEPARTMENT_NAME	COUNT(*)
10	Administration	1
20	Marketing	2
110	Accounting	2

    b. Has the highest number of employees:

DEPARTMENT_ID	DEPARTMENT_NAME	COUNT(*)
50	Shipping	5

    c. Has the lowest number of employees:

DEPARTMENT_ID	DEPARTMENT_NAME	COUNT(*)
10	Administration	1

18. Show the employee number, last name, salary, department number, and the average salary in their department for all employees.

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	AVG(S.SALARY)
100	King	90	19333.3333
101	Kochhar	90	19333.3333
102	De Haan	90	19333.3333
103	Hunold	60	6400
104	Ernst	60	6400
107	Lorentz	60	6400
124	Mourgos	50	3500
141	Rajs	50	3500
142	Davies	50	3500
143	Matos	50	3500
144	Vargas	50	3500
149	Zlotkey	80	10033.3333
174	Abel	80	10033.3333
176	Taylor	80	10033.3333
EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	AVG(S.SALARY)
200	Whalen	10	4400
201	Hartstein	20	9500
202	Fay	20	9500
205	Higgins	110	10150
206	Gietz	110	10150

19 rows selected.

19. Show all employees who were hired on the day of the week on which the highest number of employees has been hired.

LAST_NAME	DAY
Ernst	TUESDAY
Mourgos	TUESDAY
Rajs	TUESDAY
Taylor	TUESDAY
Higgins	TUESDAY
Gietz	TUESDAY

6 rows selected.

20. Create an anniversary overview based on the hire date of the employees. Sort the anniveraries in ascending order.

LAST_NAME	BIRTHDAY
Hunold	January 03
De Haan	January 13
Davies	January 29
Zlotkey	January 29
Lorentz	February 07
Hartstein	February 17
Matos	March 15
Taylor	March 24
Abel	May 11
Ernst	May 21
Grant	May 24
Higgins	June 07
Gietz	June 07
King	June 17
LAST_NAME	BIRTHDAY
Vargas	July 09
Fay	August 17
Whalen	September 17
Kochhar	September 21
Rajs	October 17
Mourgos	November 16

20 rows selected.

These exercises can be used for extra practice after you have discussed using SET operators in Lesson 15.

21. Find the job that was filled in the first half of 1990 and the same job that was filled during the same period in 1991.

JOB_ID
IT_PROG

22. Write a compound query to produce a list of employees showing raise percentages, employee IDs, and old and new salaries. Employees in departments 10, 50, and 110 are given a 5% raise, employees in department 60 are given a 10% raise, employees in departments 20 and 80 are given a 15% raise, and employees in department 90 are not given a raise.

RAISE	EMPLOYEE_ID	SALARY	NEW_SALARY
05% raise	124	5800	290
05% raise	141	3500	175
05% raise	142	3100	155
05% raise	143	2600	130
05% raise	144	2500	125
05% raise	200	4400	220
05% raise	205	12000	600
05% raise	206	8300	415
10% raise	103	9000	900
10% raise	104	6000	600
10% raise	107	4200	420
15% raise	149	10500	1575
15% raise	174	11000	1650
15% raise	176	8600	1290
15% raise	201	13000	1950
15% raise	202	6000	900
no raise	100	24000	24000
no raise	101	17000	17000
no raise	102	17000	17000

19 rows selected.

These exercises can be used for extra practice after you have discussed Oracle9*i* single row functions in Lesson 16.

23. Alter the session to set the `NLS_DATE_FORMAT` to `DD-MON-YYYY HH24:MI:SS`.

24. a. Write queries to display the time zone offsets (`TZ_OFFSET`) for the following time zones.

–Australia/Sydney

TZ_OFFS
+11:00

–Chile/EasterIsland

TZ_OFFS
-05:00

b. Alter the session to set the `TIME_ZONE` parameter value to the time zone offset of Australia/Sydney.

c. Display the `SYSDATE`, `CURRENT_DATE`, `CURRENT_TIMESTAMP`, and `LOCALTIMESTAMP` for this session. **Note**: The output might be different based on the date when the command is executed.

SYSDATE	CURRENT_DATE	CURRENT_TIMESTAMP	LOCALTIMESTAMP
09-MAR-2001 11:18:46	09-MAR-2001 16:48:46	09-MAR-01 04.48.46.183047 PM +11:00	09-MAR-01 04.48.46.183047 PM

d. Alter the session to set the `TIME_ZONE` parameter value to the time zone offset of Chile/EasterIsland.

Note: The results of the preceding question are based on a different date and in some cases they will not match the actual results that the students get. Also the time zone offset of the various countries might differ based on daylight savings time.

e. Display the SYSDATE, CURRENT_DATE, CURRENT_TIMESTAMP, and LOCALTIMESTAMP for this session. **Note**: The output might be different based on the date when the command is executed.

SYSDATE	CURRENT_DATE	CURRENT_TIMESTAMP	LOCALTIMESTAMP
09-MAR-2001 11:20:26	09-MAR-2001 00:50:27	09-MAR-01 12.50.26.718257 AM -05:00	09-MAR-01 12.50.26.718257 AM

Note: Observe in the preceding question that CURRENT_DATE, CURRENT_TIMESTAMP, and LOCALTIMESTAMP are all sensitive to the session time zone. Observe that SYSDATE is not sensitive to the session time zone.

Note: The results of the preceding question are based on a different date, and in some cases they will not match the actual results that the students get. Also the time zone offset of the various countries might differ based on daylight savings time.

25. Write a query to display the last names, month of the date of join, and hire date of those employees who have joined in the month of January, irrespective of the year of join.

LAST_NAME	EXTRACT(MONTHFROMHIRE_DATE)	HIRE_DATE
De Haan	1	13-JAN-1993 00:00:00
Hunold	1	03-JAN-1990 00:00:00
Davies	1	29-JAN-1997 00:00:00
Zlotkey	1	29-JAN-2000 00:00:00

These exercises can be used for extra practice after you have discussed enhacements to the GROUP BY clause in Lesson 17.

26. Write a query to display the following for those departments whose department ID is greater than 80 :
    - The total salary for every job within a department
    - The total salary
    - The total salary for those cities in which the departments are located
    - The total salary for every job, irrespective of the department
    - The total salary for every department irrespective of the city
    - The total salary of the cities in which the departments are located
    - Total salary for the departments, irrespective of job titles and cities

CITY	DNAME	JOB	SUM(SALARY)
Seattle	Accounting	AC_ACCOUNT	$33,200.00
Seattle	Accounting	AC_MGR	$48,000.00
Seattle	Accounting		$81,200.00
Seattle	Executive	AD_PRES	$96,000.00
Seattle	Executive	AD_VP	$1,36,000.00
Seattle	Executive		$2,32,000.00
Seattle		AC_ACCOUNT	$33,200.00
Seattle		AC_MGR	$48,000.00
Seattle		AD_PRES	$96,000.00
Seattle		AD_VP	$1,36,000.00
Seattle			$3,13,200.00
	Accounting	AC_ACCOUNT	$33,200.00
	Accounting	AC_MGR	$48,000.00
	Accounting		$81,200.00
	Executive	AD_PRES	$96,000.00
	Executive	AD_VP	$1,36,000.00
	Executive		$2,32,000.00
		AC_ACCOUNT	$33,200.00
		AC_MGR	$48,000.00
		AD_PRES	$96,000.00
		AD_VP	$1,36,000.00
			$3,13,200.00

22 rows selected.

27. Write a query to display the following groupings :

– Department ID, Job ID

– Job ID, Manager ID

The query should calculate the maximum and minimum salaries for each of these groups.

DEPARTMENT_ID	JOB	MANAGER_ID	MAX(SALARY)	MIN(SALARY)
10	AD_ASST		4400	4400
20	MK_MAN		13000	13000
20	MK_REP		6000	6000
50	ST_CLERK		3500	2500
50	ST_MAN		5800	5800
60	IT_PROG		9000	4200
80	SA_MAN		10500	10500
80	SA_REP		11000	8600
90	AD_PRES		24000	24000
90	AD_VP		17000	17000
110	AC_ACCOUNT		8300	8300
110	AC_MGR		12000	12000
	SA_REP		7000	7000
	AC_ACCOUNT	205	8300	8300
	AC_MGR	101	12000	12000
	AD_ASST	101	4400	4400
	AD_PRES		24000	24000
	AD_VP	100	17000	17000
	IT_PROG	102	9000	9000
	IT_PROG	103	6000	4200
	MK_MAN	100	13000	13000
	MK_REP	201	6000	6000
	SA_MAN	100	10500	10500
	SA_REP	149	11000	7000
	ST_CLERK	124	3500	2500
	ST_MAN	100	5800	5800

26 rows selected.

These exercises can be used for extra practice after you have discussed advanced subqueries in Lesson 18.

28. Write a query to display the top three earners in the EMPLOYEES table. Display their last names and salaries.

LAST_NAME	SALARY
King	24000
Kochhar	17000
De Haan	17000

29. Write a query to display the employee ID and last names of the employees who work in the state of California.

    Hint: Use scalar subqueries.

EMPLOYEE_ID	LAST_NAME
124	Mourgos
141	Rajs
142	Davies
143	Matos
144	Vargas

30. Write a query to delete the oldest JOB_HISTORY row of an employee by looking up the JOB_HISTORY table for the MIN(START_DATE) for the employee. Delete the records of **only** those employees who have changed at least 2 jobs. If your query has executed correctly, you will get the feedback:

```
3 rows deleted.
```

    Hint: Use a correlated DELETE.

31. Rollback the transaction.

32. Write a query to display the job ids of those jobs whose maximum salary is above half the maximum salary in the whole company. Use the WITH clause to write this query. Name the query as MAX_SAL_CALC.

```
JOB_TITLE                           JOB_TOTAL
----------------------------------- ----------
President                                24000
Administration Vice President           17000
Marketing Manager                        13000
```

These exercises can be used for extra practice after you have discussed hierarchial retrieval in Lesson 19.

33. Write a SQL statement to display employee number, last name, start date, and salary, showing:

    a. De Haan's direct reports

EMPLOYEE_ID	LAST_NAME	HIRE_DATE	SALARY
103	Hunold	03-JAN-1990 00:00:00	9000

    b. The organization tree under De Haan's (employee number 102)

EMPLOYEE_ID	LAST_NAME	HIRE_DATE	SALARY
103	Hunold	03-JAN-1990 00:00:00	9000
104	Ernst	21-MAY-1991 00:00:00	6000
107	Lorentz	07-FEB-1999 00:00:00	4200

34. Write a hierarchical query to display the employee number, manager number, and employee last name for all employees who are two levels below employee De Haan (employee number 102). Also display the level of the employee.

EMPLOYEE_ID	MANAGER_ID	LEVEL	LAST_NAME
104	103	3	Ernst
107	103	3	Lorentz

35. Produce a hierarchical report to display the employee number, manager number, the LEVEL pseudocolumn, and employee last name. For every row in the EMPLOYEES table, you should print a tree structure showing the employee, the employee's manager, then the manager's manager, and so on. Use indentations for the NAME column.

EMPLOYEE_ID	MANAGER_ID	LEVEL	LPAD(LAST_NAME,LENGTH(LAST_NAME)+(LEVEL*2)-2,'_')
100		1	King
101	100	1	Kochhar
100		2	__King
102	100	1	De Haan
100		2	__King
103	102	1	Hunold
102	100	2	__De Haan
100		3	____King
104	103	1	Ernst
100		4	______King
107	103	1	Lorentz
103	102	2	__Hunold
201	100	2	__Hartstein
100		3	____King
205	101	1	Higgins
101	100	2	__Kochhar
100		3	____King
206	205	1	Gietz
205	101	2	__Higgins
101	100	3	____Kochhar
100		4	______King

56 rows selected.

Note: The output shown is only a sample. All the rows from the actual output are not included here.

These exercises can be used for extra practice after you have discussed Oracle 9*i* extensions to DML and DDL statements in Lesson 20.

36. Write a query to do the following:
    - Retrieve the details of the employee ID, hire date, salary, and manager ID of those employees whose employee ID is more than or equal to 200 from the `EMPLOYEES` table.
    - If the salary is less than $5,000, insert the details of employee ID and salary into the `SPECIAL_SAL` table.
    - Insert the details of employee ID, hire date, and salary into the `SAL_HISTORY` table.
    - Insert the details of employee ID, manager ID, and salary into the `MGR_HISTORY` table.

37. Query the `SPECIAL_SAL`, `SAL_HISTORY` and the `MGR_HISTORY` tables to view the inserted records.

`SPECIAL_SAL` Table

EMPLOYEE_ID	SALARY
200	4400

`SAL_HISTORY` Table

EMPLOYEE_ID	HIRE_DATE	SALARY
201	17-FEB-1996 00:00:00	13000
202	17-AUG-1997 00:00:00	6000
205	07-JUN-1994 00:00:00	12000
206	07-JUN-1994 00:00:00	8300

`MGR_HISTORY` Table

EMPLOYEE_ID	MANAGER_ID	SALARY
201	100	13000
202	201	6000
205	101	12000
206	205	8300

38. Create the `LOCATIONS_NAMED_INDEX` table based on the following table instance chart. Name the index for the `PRIMARY KEY` column as `LOCATIONS_PK_IDX`.

COLUMN Name	**Deptno**	**Dname**
Primary Key	Yes	
Datatype	Number	VARCHAR2
Length	4	30

39. Query the `USER_INDEXES` table to display the `INDEX_NAME` for the `LOCATIONS_NAMED_INDEX` table.

INDEX_NAME	TABLE_NAME
LOCATIONS_PK_IDX	LOCATIONS_ NAMED_INDE X

This exercise can be used for extra practice after you have discussed writing advanced scripts in Appendix D.

Appendix D Additional Practice

Write a SQL script file to drop all objects (tables, views, indexes, sequences, synonyms, and so on) that you own. The output shown is only a guideline.

```
DROP INDEX COUNTRY_C_ID_PK;
DROP INDEX DEPT_ID_PK;
DROP INDEX DEPT_LOCATION_IX;
DROP INDEX EMP_EMAIL_UK;
DROP INDEX EMP_MANAGER_IX;
DROP INDEX EMP_JOB_IX;
DROP INDEX EMP_EMP_ID_PK;
DROP INDEX EMP_DEPARTMENT_IX;
DROP INDEX EMP_NAME_IX;
DROP INDEX REG_ID_PK;
DROP INDEX LOC_STATE_PROVINCE_IX;
```

```
OP TABLE HIREDATE_HISTORY_00;
DROP TABLE HIREDATE_HISTORY_99;
DROP TABLE HIREDATE_HISTORY;
DROP TABLE EMPHISTORY;
DROP TABLE EMPLOYEES;
DROP VIEW EMP_DETAILS_VIEW;
```

Additional Practice Solutions

These exercises can be used for extra practice after you have discussed the following topics: basic SQL SELECT statement, basic *i*SQL*Plus commands, and SQL functions.

1. Show all data of the clerks who have been hired after the year 1997.

```
SELECT *
FROM   employees
WHERE  job_id = 'ST_CLERK'
AND hire_date > '31-DEC-1997';
```

2. Show the last name, job, salary, and commission of those employees who earn commission. Sort the data by the salary in descending order.

```
SELECT last_name, job_id, salary, commission_pct
FROM   employees
WHERE  commission_pct IS NOT NULL
ORDER BY salary DESC;
```

3. Show the employees that have no commission with a 10% raise intheir salary (round off the salaries).

```
SELECT 'The salary of '||last_name||' after a 10% raise is '
        || ROUND(salary*1.10) "New salary"
FROM   employees
WHERE  commission_pct IS NULL;
```

4. Show the last names of all employees together with the number of years and the number of completed months that they have been employed.

```
SELECT last_name,
    TRUNC(MONTHS_BETWEEN(SYSDATE, hire_date) / 12) YEARS,
    TRUNC(MOD(MONTHS_BETWEEN(SYSDATE, hire_date), 12)) MONTHS
FROM           employees;
```

5. Show those employees that have a name starting with *J*, *K*, *L*, or *M*.

```
SELECT last_name
FROM   employees
WHERE  SUBSTR(last_name, 1,1) IN ('J', 'K', 'L', 'M');
```

6. Show all employees, and indicate with "Yes" or "No" whether they receive a commission.

```
SELECT last_name, salary,
       decode(commission_pct, NULL, 'No', 'Yes') commission
FROM   employees;
```

These exercises can be used for extra practice after you have discussed the following topics: SQL basic SELECT statement, basic *i*SQL*Plus commands, SQL functions, joins, and group functions.

7. Show the department names, locations, names, job titles, and salaries of employees who work in location 1800.

```
SELECT d.department_name, d.location_id,
       e.last_name, e.job_id, e.salary
FROM   employees e, departments d
WHERE   e.department_id = d.department_id
AND     d.location_id = 1800;
```

8. How many employees have a name that ends with an *n*? Create two possible solutions.

```
SELECT COUNT(*)
FROM   employees
WHERE  last_name LIKE '%n';

SELECT COUNT(*)
FROM   employees
WHERE  SUBSTR(last_name, -1) = 'n';
```

9. Show the names and locations for all departments and the number of employees working in each department. Make sure that departments without employees are included, as well.

```
SELECT d.department_id, d.department_name,
       d.location_id,   COUNT(e.employee_id)
FROM   employees e, departments d
WHERE   e.department_id(+) = d.department_id
GROUP BY d.department_id, d.department_name, d.location_id;
```

10. Which jobs are found in departments 10 and 20?

```
SELECT DISTINCT job_id
FROM   employees
WHERE  department_id IN (10, 20);
```

11. Which jobs are found in the Administration and Executive departments, and how many employees do these jobs? Show the job with the highest frequency first.

```
SELECT e.job_id, count(e.job_id) FREQUENCY
FROM    employees e, departments d
WHERE  e.department_id = d.department_id
AND    d.department_name IN ('Administration', 'Executive')
GROUP BY e.job_id
ORDER BY FREQUENCY DESC;
```

These exercises can be used for extra practice after you have discussed the following topics: basic SQL SELECT statements, basic *i*SQL*Plus commands, SQL functions, joins, group functions, subqueries.

12. Show all employees who were hired in the first half of the month (before the 16th of the month).

```
SELECT last_name, hire_date
FROM   employees
WHERE  TO_CHAR(hire_date, 'DD') < 16;
```

13. Show the names, salaries, and the number of dollars (in thousands) that all employees earn.

```
SELECT last_name, salary, TRUNC(salary, -3)/1000 Thousands
FROM   employees;
```

14. Show all employees who have managers with a salary higher than $15,000. Show the following data: employee name, manager name, manager salary, and salary grade of the manager.

```
SELECT e.last_name, m.last_name manager, m.salary,
       j.grade_level
FROM   employees e, employees m, job_grades j
WHERE  e.manager_id = m.employee_id
AND    m.salary BETWEEN j.lowest_sal AND j.highest_sal
AND    m.salary > 15000;
```

15. Show the department number, name, number of employees, and average salary of all departments, together with the names, salaries, and jobs of the employees working in each department.

```
BREAK ON department_id -
  ON department_name ON employees ON avg_sal SKIP 1

SELECT  d.department_id, d.department_name,
        count(e1.employee_id) employees,
        NVL(TO_CHAR(AVG(e1.salary), '99999.99'),
        'No average' ) avg_sal,
        e2.last_name, e2.salary, e2.job_id
FROM    departments d, employees e1, employees e2
WHERE   d.department_id = e1.department_id(+)
AND     d.department_id = e2.department_id(+)
GROUP BY d.department_id, d.department_name,
         e2.last_name,   e2.salary, e2.job_id
ORDER BY d.department_id, employees;

CLEAR BREAKS
```

16. Show the department number and the lowest salary of the department with the highest average salary.

```
SELECT department_id, MIN(salary)
FROM   employees
GROUP BY department_id
HAVING AVG(salary) = (SELECT MAX(AVG(salary))
                      FROM   employees
                      GROUP BY department_id);
```

17. Show the department numbers, names, and locations of the departments where no sales representatives work.

```
SELECT *
FROM   departments
WHERE  department_id NOT IN(SELECT department_id
                            FROM employees
                            WHERE job_id = 'SA_REP'
                            AND department_id IS NOT NULL);
```

18. Show the department number, department name, and the number of employees working in each that:

    a. Includes fewer than 3 employees:

```
SELECT d.department_id, d.department_name, COUNT(*)
FROM   departments d, employees e
WHERE  d.department_id = e.department_id
GROUP BY d.department_id, d.department_name
HAVING COUNT(*) < 3;
```

b. Has the highest number of employees:

```
SELECT d.department_id, d.department_name, COUNT(*)
FROM   departments d, employees e
WHERE  d.department_id = e.department_id
GROUP BY d.department_id, d.department_name
HAVING COUNT(*) = (SELECT MAX(COUNT(*))
                   FROM   employees
                   GROUP BY department_id);
```

c. Has the lowest number of employees:

```
SELECT d.department_id, d.department_name, COUNT(*)
FROM   departments d, employees e
WHERE  d.department_id = e.department_id
GROUP BY d.department_id, d.department_name
HAVING COUNT(*) = (SELECT MIN(COUNT(*))
                   FROM   employees
                   GROUP BY department_id);
```

19. Show the employee number, last name, salary, department number, and the average salary in their department for all employees.

```
SELECT e.employee_id, e.last_name,
       e.department_id,   AVG(s.salary)
FROM   employees e, employees s
WHERE   e.department_id = s.department_id
GROUP BY e.employee_id, e.last_name, e.department_id;
```

20. Show all employees who were hired on the day of the week on which the highest number of employees has been hired.

```
SELECT last_name, TO_CHAR(hire_date, 'DAY') day
FROM   employees
WHERE  TO_CHAR(hire_date, 'Day') =
    (SELECT TO_CHAR(hire_date, 'Day')
     FROM   employees
     GROUP BY TO_CHAR(hire_date, 'Day')
     HAVING COUNT(*) = (SELECT MAX(COUNT(*))
                        FROM   employees
                        GROUP BY TO_CHAR(hire_date, 'Day')));
```

21. Create an anniversary overview based on the hire date of the employees. Sort the anniveraries in ascending order.

```
SELECT last_name, TO_CHAR(hire_date, 'Month DD') BIRTHDAY
FROM   employees
ORDER BY TO_CHAR(hire_date, 'DDD');
```

These exercises can be used for extra practice after you have discussed using SET operators in Lesson 15.

22. Find the job that was filled in the first half of 1990 and the same job that was filled during the same period in 1991.

```
SELECT job_id
FROM   employees
WHERE  hire_date
BETWEEN '01-JAN-1990' AND '30-JUN-1990'
INTERSECT
SELECT job_id
FROM   employees
WHERE  hire_date BETWEEN '01-JAN-1991'
AND '30-JUN-1991';
```

23. Write a compound query to produce a list of employees showing raise percentages, employee IDs, and old and new salaries. Employees in departments 10, 50, and 110 are given a 5% raise, employees in department 60 are given a 10% raise, employees in departments 20 and 80 are given a 15% raise, and employees in department 90 are not given a raise.

```
SELECT '05% raise' raise, employee_id, salary,
        salary *.05 new_salary
FROM   employees
WHERE  department_id IN (10,50, 110)
UNION
SELECT '10% raise', employee_id, salary, salary * .10
FROM   employees
WHERE  department_id = 60
UNION
SELECT '15% raise', employee_id, salary, salary * .15
FROM   employees
WHERE  department_id IN (20, 80)
UNION
SELECT 'no raise', employee_id, salary, salary
FROM   employees
WHERE  department_id = 90;
```

These exercises can be used for extra practice after you have discussed Oracle9*i* single row functions in Lesson 16.

24. Alter the session to set the NLS_DATE_FORMAT to DD-MON-YYYY HH24:MI:SS.

```
ALTER SESSION
SET NLS_DATE_FORMAT = 'DD-MON-YYYY HH24:MI:SS';
```

25. a. Write queries to display the time zone offsets (TZ_OFFSET), for the following time zones.

–Australia/Sydney

```
TZ_OFFS
-------
+11:00
SELECT TZ_OFFSET ('Australia/Sydney') from dual;
```

–Chile/EasterIsland

```
TZ_OFFS
-------
-05:00
SELECT TZ_OFFSET ('Chile/EasterIsland') from dual;
```

b. Alter the session to set the TIME_ZONE parameter value to the time zone offset of Australia/Sydney.

```
ALTER SESSION SET TIME_ZONE = '+11:00';
```

c. Display the SYSDATE, CURRENT_DATE, CURRENT_TIMESTAMP, and LOCALTIMESTAMP for this session. **Note**: The output might be different based on the date when the command is executed.

```
SELECT SYSDATE,CURRENT_DATE,
CURRENT_TIMESTAMP, LOCALTIMESTAMP
FROM DUAL;
```

d. Alter the session to set the TIME_ZONE parameter value to the time zone offset of Chile/EasterIsland.

```
ALTER SESSION SET TIME_ZONE = '-05:00';
```

e. Display the SYSDATE, CURRENT_DATE, CURRENT_TIMESTAMP, and LOCALTIMESTAMP for this session. **Note**: The output might be different based on the date when the command is executed.

```
SELECT SYSDATE,CURRENT_DATE,
CURRENT_TIMESTAMP, LOCALTIMESTAMP
FROM DUAL;
```

Note: Observe in the preceding question that CURRENT_DATE, CURRENT_TIMESTAMP, and LOCALTIMESTAMP are all sensitive to the session time zone. Observe that SYSDATE is not sensitive to the session time zone.

Note: The results of the preceding question are based on a different date, and in some cases they will not match the actual results that the students get. Also the time zone offset of the various countries might differ based on daylight savings time.

26. Write a query to display the last names, month of the date of join, and hire date of those employees who have joined in the month of January, irrespective of the year of join.

```
SELECT last_name, EXTRACT (MONTH FROM HIRE_DATE),HIRE_DATE
FROM employees
WHERE EXTRACT (MONTH FROM HIRE_DATE) = 1;
```

These exercises can be used for extra practice after you have discussed enhacements to the GROUP BY clause in Lesson 17.

27. Write a query to display the following for those departments whose department ID is greater than 80:
    - The total salary for every job within a department
    - The total salary
    - The total salary for those cities in which the departments are located
    - The total salary for every job, irrespective of the department
    - The total salary for every department irrespective of the city
    - The total salary of the cities in which the departments are located
    - Total salary for the departments, irrespective of job titles and cities

```
COLUMN city FORMAT A25 Heading CITY
COLUMN department_name FORMAT A15 Heading DNAME
COLUMN job_id  FORMAT A10 Heading JOB
COLUMN SUM(salary)  FORMAT $99,99,999.00 Heading SUM(SALARY)

SELECT   city,department_name, job_id, SUM(salary)
FROM     countries,locations,employees,departments
WHERE    departments.location_id = locations.location_id
AND      employees.department_id = departments.department_id
AND      employees.department_id > 80
GROUP    BY CUBE( city,department_name, job_id);
```

28. Write a query to display the following groupings :
    - Department ID, Job ID
    - Job ID, Manager ID

    The query should calculate the maximum and minimum salaries for each of these groups.

```
SELECT department_id, job_id, manager_id,max(salary),min(salary)
FROM   employees
GROUP BY GROUPING SETS
((department_id,job_id), (job_id,manager_id));
```

These exercises can be used for extra practice after you have discussed advanced subqueries in Lesson 18.

29. Write a query to display the top three earners in the EMPLOYEES table. Display their last names and salaries.

```
SELECT last_name, salary
FROM   employees e
WHERE  3  > (SELECT COUNT(*)
             FROM   employees
             WHERE  e.salary < salary);
```

30. Write a query to display the employee ID and last names of the employees who work in the state of California.

    Hint: Use scalar subqueries.

```
SELECT employee_id, last_name
FROM employees e
WHERE ((SELECT location_id
        FROM departments d
        WHERE e.department_id = d.department_id )
              IN   (SELECT location_id
                    FROM locations l
                    WHERE STATE_province = 'California'));
```

31. Write a query to delete the oldest JOB_HISTORY row of an employee by looking up the JOB_HISTORY table for the MIN(START_DATE) for the employee. Delete the records of *only* those employees who have changed at least 2 jobs. If your query has executed correctly, you will get the following feedback:

    Hint: Use a correlated DELETE.

```
DELETE FROM job_history JH
WHERE employee_id =
                  (SELECT employee_id
                   FROM employees E
                   WHERE JH.employee_id = E.employee_id
                   AND START_DATE = (SELECT MIN(start_date)
                            FROM job_history JH
                    WHERE JH.employee_id = E.employee_id)
                            AND 3 >  (SELECT COUNT(*)
                            FROM job_history JH
                             WHERE JH.employee_id = E.employee_id
                             GROUP BY EMPLOYEE_ID
                             HAVING COUNT(*) >= 2));
```

32. Rollback the transaction.

```
ROLLBACK;
```

33. Write a query to display the job IDs of those jobs whose maximum salary is above half the maximum salary in the whole company. Use the `WITH` clause to write this query. Name the query as `MAX_SAL_CALC`.

```
WITH
MAX_SAL_CALC AS (
SELECT job_title, MAX(salary) AS job_total
FROM employees, jobs
WHERE employees.job_id = jobs.job_id
GROUP BY job_title)
SELECT job_title, job_total
FROM MAX_SAL_CALC
WHERE job_total > (SELECT MAX(job_total) * 1/2
                   FROM MAX_SAL_CALC)
ORDER BY job_total DESC;
```

These exercises can be used for extra practice after you have discussed hierarchial retrieval in Lesson 19.

34. Write a SQL statement to display employee number, last name, start date, and salary, showing:

a. De Haan's direct reports

```
SELECT employee_id, last_name, hire_date, salary
FROM   employees
WHERE  manager_id = (SELECT employee_id
                     FROM   employees
                     WHERE last_name = 'De Haan');
```

b. The organization tree under De Haan's (employee number 102)

```
SELECT employee_id, last_name, hire_date, salary
 FROM   employees
 WHERE  employee_id != 102
 CONNECT BY manager_id = PRIOR employee_id
 START WITH employee_id = 102;
```

35. Write a hierarchical query to display the employee number, manager number, and mployee last name for all employees who are two levels below employee De Haan (employee number 102). Also display the level of the employee.

```
SELECT employee_id, manager_id, level, last_name
 FROM   employees
 WHERE LEVEL = 3
 CONNECT BY manager_id = PRIOR employee_id
 START WITH employee_id= 102;
```

36. Produce a hierarchical report to display employee number, manager number, the LEVEL pseudocolumn, and employee last name. For every row in the EMPLOYEES table, you should print a tree structure showing the employee, the employee's manager, then the manager's manager, and so on. Use indentations for the NAME column.

```
COLUMN name FORMAT A25
SELECT  employee_id, manager_id, LEVEL,
LPAD(last_name, LENGTH(last_name)+(LEVEL*2)-2,'_')
FROM    employees
CONNECT BY employee_id = PRIOR manager_id;
COLUMN name CLEAR
```

These exercises can be used for extra practice after you have discussed Oracle 9*i* extensions to DML and DDL statements in Lesson 20.

37. Write a query to do the following:
    - Retrieve the details of the employee ID, hire date, salary, and manager ID of those employees whose employee ID is more than or equal to 200 from the EMPLOYEES table.
    - If the salary is less than $5,000, insert the details of employee ID and salary into the SPECIAL_SAL table.
    - Insert the details of employee ID, hire date, and salary into the SAL_HISTORY table.
    - Insert the details of employee ID, manager ID, and salary into the MGR_HISTORY table.

```
INSERT ALL
WHEN SAL < 5000 THEN
INTO  special_sal VALUES (EMPID, SAL)
ELSE
INTO sal_history VALUES(EMPID,HIREDATE,SAL)
INTO mgr_history VALUES(EMPID,MGR,SAL)
SELECT employee_id EMPID, hire_date HIREDATE,
       salary SAL, manager_id MGR
FROM employees
WHERE employee_id >=200;
```

38. Query the SPECIAL_SAL, SAL_HISTORY and the MGR_HISTORY tables to view the inserted records.

```
SELECT * FROM special_sal;
SELECT * FROM sal_history;
SELECT * FROM mgr_history;
```

39. Create the LOCATIONS_NAMED_INDEX table based on the following table instance chart. Name the index for the PRIMARY KEY column as LOCATIONS_PK_IDX.

```
CREATE TABLE LOCATIONS_NAMED_INDEX
(location_id NUMBER(4) PRIMARY KEY USING INDEX
(CREATE INDEX locations_pk_idx ON
LOCATIONS_NAMED_INDEX(location_id)),
location_name VARCHAR2(20));
```

40. Query the USER_INDEXES table to display the INDEX_NAME for the LOCATIONS_NAMED_INDEX table.

```
SELECT INDEX_NAME, TABLE_NAME
FROM USER_INDEXES
WHERE TABLE_NAME = 'LOCATIONS_NAMED_INDEX';
```

This exercise can be used for extra practice after you have discussed writing advanced scripts in Appendix D.

Appendix D Additional Practice

Write a SQL script file to drop all objects (tables, views, indexes, sequences, synonyms, and so on) that you own. The output shown is only a guideline.

```
SET HEADING OFF ECHO OFF FEEDBACK OFF TERMOUT OFF
SET PAGESIZE 0

SPOOL dropall.sql

SELECT   'DROP ' || object_type || ' ' || object_name || ';'
FROM     user_objects
ORDER BY object_type
/

SPOOL off

SET HEADING ON ECHO ON FEEDBACK ON TERMOUT ON
SET PAGESIZE 24
```

Table Descriptions and Data

COUNTRIES Table

```
DESCRIBE countries
```

Name	Null?	Type
COUNTRY_ID	NOT NULL	CHAR(2)
COUNTRY_NAME		VARCHAR2(40)
REGION_ID		NUMBER

```
SELECT * FROM countries;
```

CO	COUNTRY_NAME	REGION_ID
CA	Canada	2
DE	Germany	1
UK	United Kingdom	1
US	United States of America	2

DEPARTMENTS Table

```
DESCRIBE departments
```

Name	Null?	Type
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)

```
SELECT * FROM departments;
```

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500
90	Executive	100	1700
110	Accounting	205	1700
190	Contracting		1700

8 rows selected.

EMPLOYEES Table

```
DESCRIBE employees
```

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

```
SELECT * FROM employees;
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SA
100	Steven	King	SKING	515.123.4567	17-JUN-87	AD_PRES	
101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	
102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	
103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-90	IT_PROG	
104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-91	IT_PROG	
107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-99	IT_PROG	
124	Kevin	Mourgos	KMOURGOS	650.123.5234	16-NOV-99	ST_MAN	
141	Trenna	Rajs	TRAJS	650.121.8009	17-OCT-95	ST_CLERK	
142	Curtis	Davies	CDAVIES	650.121.2994	29-JAN-97	ST_CLERK	
143	Randall	Matos	RMATOS	650.121.2874	15-MAR-98	ST_CLERK	
144	Peter	Vargas	PVARGAS	650.121.2004	09-JUL-98	ST_CLERK	
149	Eleni	Zlotkey	EZLOTKEY	011.44.1344.429018	29-JAN-00	SA_MAN	
174	Ellen	Abel	EABEL	011.44.1644.429267	11-MAY-96	SA_REP	
176	Jonathon	Taylor	JTAYLOR	011.44.1644.429265	24-MAR-98	SA_REP	
178	Kimberely	Grant	KGRANT	011.44.1644.429263	24-MAY-99	SA_REP	
200	Jennifer	Whalen	JWHALEN	515.123.4444	17-SEP-87	AD_ASST	
201	Michael	Hartstein	MHARTSTE	515.123.5555	17-FEB-96	MK_MAN	
202	Pat	Fay	PFAY	603.123.6666	17-AUG-97	MK_REP	
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-JUN-94	AC_MGR	
206	William	Gietz	WGIETZ	515.123.8181	07-JUN-94	AC_ACCOUNT	

20 rows selected.

EMPLOYEES Table (continued)

	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
	24000			90
	17000		100	90
	17000		100	90
	9000		102	60
	6000		103	60
	4200		103	60
	5800		100	50
	3500		124	50
	3100		124	50
	2600		124	50
	2500		124	50
	10500	.2	100	80
	11000	.3	149	80
	8600	.2	149	80
	7000	.15	149	
	4400		101	10
	13000		100	20
	6000		201	20
	12000		101	110
T	8300		205	110

JOBS Table

```
DESCRIBE jobs
```

Name	Null?	Type
JOB_ID	NOT NULL	VARCHAR2(10)
JOB_TITLE	NOT NULL	VARCHAR2(35)
MIN_SALARY		NUMBER(6)
MAX_SALARY		NUMBER(6)

```
SELECT * FROM jobs;
```

JOB_ID	JOB_TITLE	MIN_SALARY	MAX_SALARY
AD_PRES	President	20000	40000
AD_VP	Administration Vice President	15000	30000
AD_ASST	Administration Assistant	3000	6000
AC_MGR	Accounting Manager	8200	16000
AC_ACCOUNT	Public Accountant	4200	9000
SA_MAN	Sales Manager	10000	20000
SA_REP	Sales Representative	6000	12000
ST_MAN	Stock Manager	5500	8500
ST_CLERK	Stock Clerk	2000	5000
IT_PROG	Programmer	4000	10000
MK_MAN	Marketing Manager	9000	15000
MK_REP	Marketing Representative	4000	9000

12 rows selected.

JOB_GRADES Table

```
DESCRIBE job_grades
```

Name	Null?	Type
GRADE_LEVEL		VARCHAR2(3)
LOWEST_SAL		NUMBER
HIGHEST_SAL		NUMBER

```
SELECT * FROM job_grades;
```

GRA	LOWEST_SAL	HIGHEST_SAL
A	1000	2999
B	3000	5999
C	6000	9999
D	10000	14999
E	15000	24999
F	25000	40000

6 rows selected.

JOB_HISTORY Table

```
DESCRIBE job_history
```

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
START_DATE	NOT NULL	DATE
END_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
DEPARTMENT_ID		NUMBER(4)

```
SELECT * FROM job_history;
```

EMPLOYEE_ID	START_DAT	END_DATE	JOB_ID	DEPARTMENT_ID
102	13-JAN-93	24-JUL-98	IT_PROG	60
101	21-SEP-89	27-OCT-93	AC_ACCOUNT	110
101	28-OCT-93	15-MAR-97	AC_MGR	110
201	17-FEB-96	19-DEC-99	MK_REP	20
114	24-MAR-98	31-DEC-99	ST_CLERK	50
122	01-JAN-99	31-DEC-99	ST_CLERK	50
200	17-SEP-87	17-JUN-93	AD_ASST	90
176	24-MAR-98	31-DEC-98	SA_REP	80
176	01-JAN-99	31-DEC-99	SA_MAN	80
200	01-JUL-94	31-DEC-98	AC_ACCOUNT	90

10 rows selected.

LOCATIONS Table

```
DESCRIBE locations
```

Name	Null?	Type
LOCATION_ID	NOT NULL	NUMBER(4)
STREET_ADDRESS		VARCHAR2(40)
POSTAL_CODE		VARCHAR2(12)
CITY	NOT NULL	VARCHAR2(30)
STATE_PROVINCE		VARCHAR2(25)
COUNTRY_ID		CHAR(2)

```
SELECT * FROM locations;
```

LOCATION_ID	STREET_ADDRESS	POSTAL_CODE	CITY	STATE_PROVINCE	CO
1400	2014 Jabberwocky Rd	26192	Southlake	Texas	US
1500	2011 Interiors Blvd	99236	South San Francisco	California	US
1700	2004 Charade Rd	98199	Seattle	Washington	US
1800	460 Bloor St. W.	ON M5S 1X8	Toronto	Ontario	CA
2500	Magdalen Centre, The Oxford Science Park	OX9 9ZB	Oxford	Oxford	UK

REGIONS Table

```
DESCRIBE regions
```

Name	Null?	Type
REGION_ID	NOT NULL	NUMBER
REGION_NAME		VARCHAR2(25)

```
SELECT * FROM regions;
```

REGION_ID	REGION_NAME
1	Europe
2	Americas
3	Asia
4	Middle East and Africa