Oracle Database 11*g*: SQL Fundamentals II

Additional Material

D49994GC11

Edition 1.1

May 2009

D60301

Authors

Chaitanya Koratamaddi

Brian Pottle

Technical Contributors and Reviewers

Claire Bennett
Ken Cooper
Yanti Chang
Laszlo Czinkoczki
Burt Demchick
Gerlinde Frenzen
Joel Goodman
Laura Garza
Richard Green
Nancy Greenberg
Akira Kinutani
Wendy Lo
Isabelle Marchand
Timothy Mcglue
Alan Paulson
Srinivas Putrevu
Bryan Roberts
Clinton Shaffer
Abhishek Singh
Jenny Tsai Smith
James Spiller
Lori Tritz
Lex van der Werff
Marcie Young

Editors

Nita Pavitran
Arijit Ghosh
Raj Kumar

Graphic Designer

Satish Bettegowda

Publishers

Syed Ali

Jayanthy Keshavamurthy

Contents

Additional Practices

The following exercises can be used for extra practice after you have discussed data manipulation language (DML) and data definition language (DDL) statements in the lessons titled "Managing Schema Objects" and "Manipulating Large Data Sets."

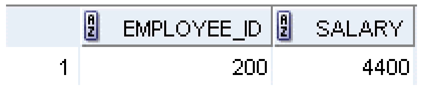
Note: Run the `lab_ap_cre_special_sal.sql`, `lab_ap_cre_sal_history.sql`, and `lab_ap_cre_mgr_history.sql` scripts in the labs folder to create the `SPECIAL_SAL`, `SAL_HISTORY`, and `MGR_HISTORY` tables.

1. The Human Resources department wants to get a list of underpaid employees, salary history of employees, and salary history of managers based on an industry salary survey. So they have asked you to do the following:

   Write a statement to do the following:

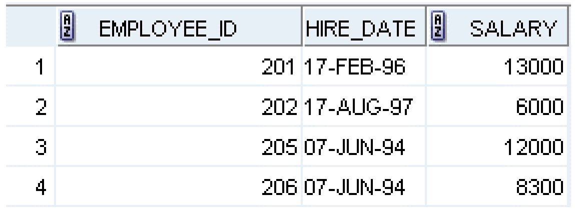
   - Retrieve details such as the employee ID, hire date, salary, and manager ID of those employees whose employee ID is more than or equal to 200 from the `EMPLOYEES` table.
   - If the salary is less than $5,000, insert details such as the employee ID and salary into the `SPECIAL_SAL` table.
   - Insert details such as the employee ID, hire date, and salary into the `SAL_HISTORY` table.
   - Insert details such as the employee ID, manager ID, and salary into the `MGR_HISTORY` table.
2. Query the `SPECIAL_SAL`, `SAL_HISTORY`, and `MGR_HISTORY` tables to view the inserted records.

`SPECIAL_SAL`



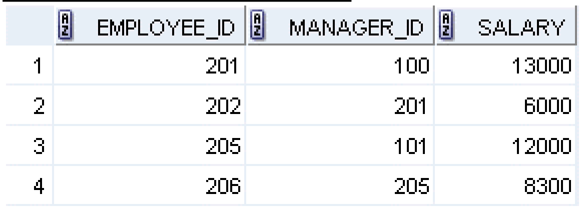
	EMPLOYEE_ID	SALARY
1	200	4400

`SALARY_HISTORY`



	EMPLOYEE_ID	HIRE_DATE	SALARY
1	201	17-FEB-96	13000
2	202	17-AUG-97	6000
3	205	07-JUN-94	12000
4	206	07-JUN-94	8300

```
MGR_HISTORY
```

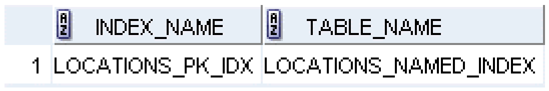


	EMPLOYEE_ID	MANAGER_ID	SALARY
1	201	100	13000
2	202	201	6000
3	205	101	12000
4	206	205	8300

3. Nita, the DBA, needs you to create a table, which has a primary key constraint, but she wants to name the index to have a different name than the constraint. 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	
Data Type	Number	VARCHAR2
Length	4	30

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



	INDEX_NAME	TABLE_NAME
1	LOCATIONS_PK_IDX	LOCATIONS_NAMED_INDEX

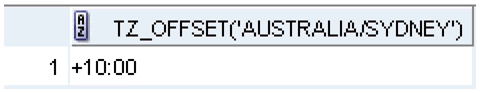
The following exercises can be used for extra practice after you have discussed datetime functions.

You work for a global company and the new vice president of operations wants to know the different time zones of all the company branches. The new vice president has requested the following information:

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

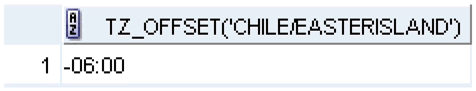
6. a. Write queries to display the time zone offsets (`TZ_OFFSET`) for the following time zones:

Australia/Sydney



	TZ_OFFSET('AUSTRALIA/SYDNEY')
1	+10:00

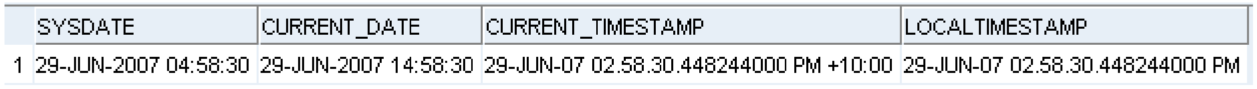
Chile/Easter Island



	TZ_OFFSET('CHILE/EASTERISLAND')
1	-06:00

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

c. Display `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.



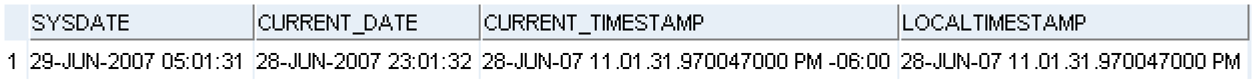
	SYSDATE	CURRENT_DATE	CURRENT_TIMESTAMP	LOCALTIMESTAMP
1	29-JUN-2007 04:58:30	29-JUN-2007 14:58:30	29-JUN-07 02.58.30.448244000 PM +10:00	29-JUN-07 02.58.30.448244000 PM

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

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 may differ, based on daylight saving time.

e. Display `SYSDATE`, `CURRENT_DATE`, `CURRENT_TIMESTAMP`, and `LOCALTIMESTAMP` for this session.

Note: The output may be different based on the date when the command is executed.



	SYSDATE	CURRENT_DATE	CURRENT_TIMESTAMP	LOCALTIMESTAMP
1	29-JUN-2007 05:01:31	28-JUN-2007 23:01:32	28-JUN-07 11.01.31.970047000 PM -06:00	28-JUN-07 11.01.31.970047000 PM

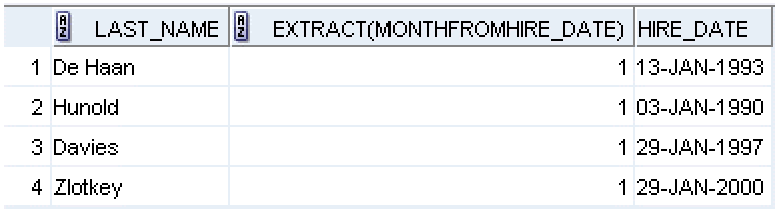
f. Alter the session to set `NLS_DATE_FORMAT` to `DD-MON-YYYY`.

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.
- 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 may differ, based on daylight saving time.

7. The Human Resources department wants a list of employees who are up for review in January, so they have requested you to do the following:

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

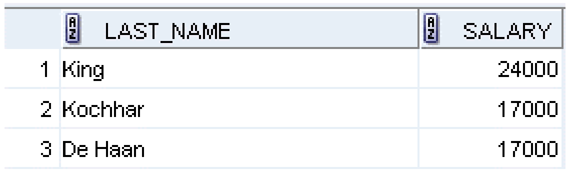


	LAST_NAME	EXTRACT(MONTHFROMHIRE_DATE)	HIRE_DATE
1	De Haan	1	13-JAN-1993
2	Hunold	1	03-JAN-1990
3	Davies	1	29-JAN-1997
4	Zlotkey	1	29-JAN-2000

The following exercises can be used for extra practice after you have discussed advanced subqueries.

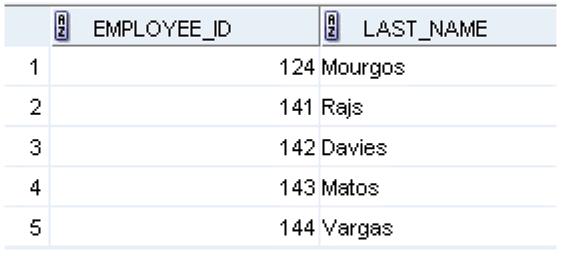
8. The CEO needs a report on the top three earners in the company for profit sharing. You are responsible to provide the CEO with a list.

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



	LAST_NAME	SALARY
1	King	24000
2	Kochhar	17000
3	De Haan	17000

9. The benefits for the state of California have been changed based on a local ordinance. So the benefits representative has asked you to compile a list of the people who are affected. 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
1	124	Mourgos
2	141	Rajs
3	142	Davies
4	143	Matos
5	144	Vargas

10. Nita, the DBA, wants to remove old information from the database. One of the things she thinks is unnecessary is the old employment records. She has asked you to do the following:

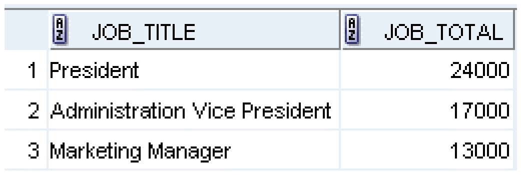
    Write a query to delete the oldest `JOB_HISTORY` row of an employee by looking up the `JOB_HISTORY` table for the `MIN(START_DATE)` for the employee. Delete the records of *only* those employees who have changed at least two jobs.
    Hint: Use a correlated `DELETE` command.

11. The vice president of Human Resources needs the complete employment records for the annual employee recognition banquet speech. The vice president makes a quick phone call to stop you from following the DBA's orders.

    Roll back the transaction.

12. The sluggish economy is forcing management to take cost reduction actions. The CEO wants to review the highest paid jobs in the company. You are responsible to provide the CEO with a list based on the following specifications:

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

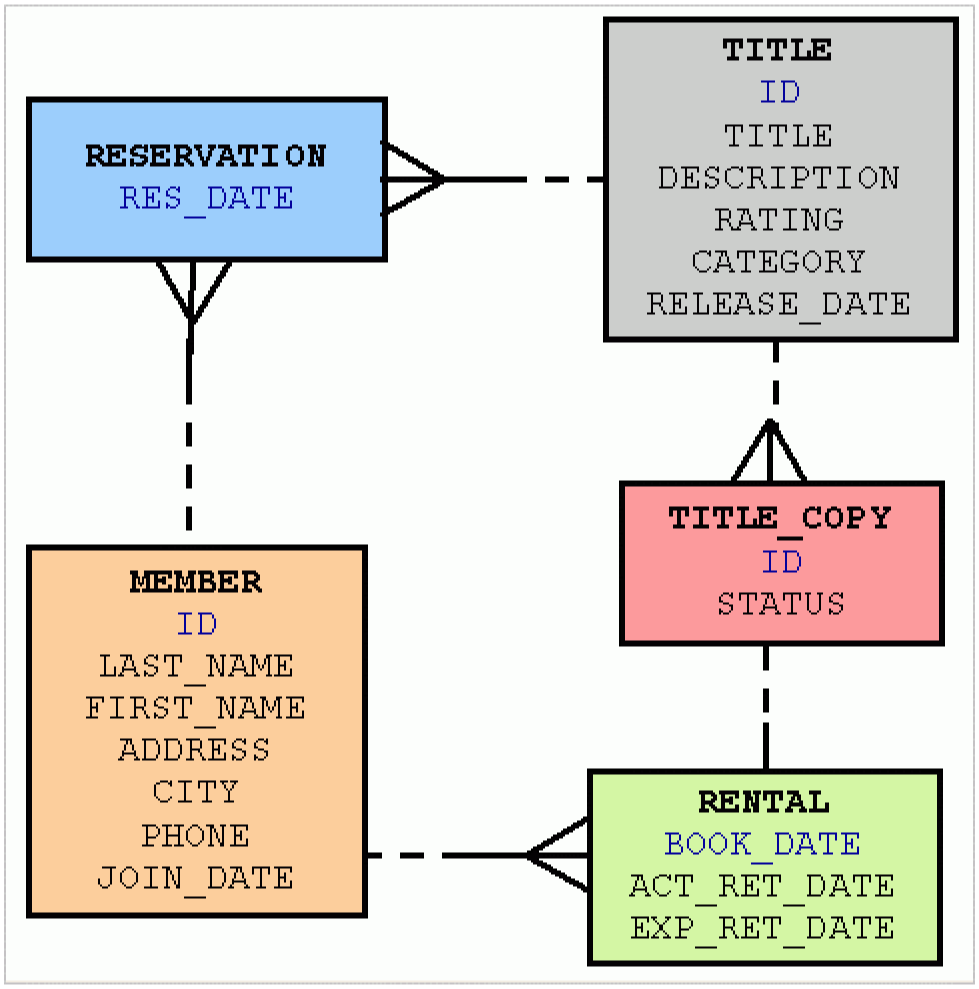


	JOB_TITLE	JOB_TOTAL
1	President	24000
2	Administration Vice President	17000
3	Marketing Manager	13000

Additional Practices: Case Study

In the case study for the *SQL Fundamentals I* course, you built a set of database tables for a video application. In addition, you inserted, updated, and deleted records in a video store database and generated a report.

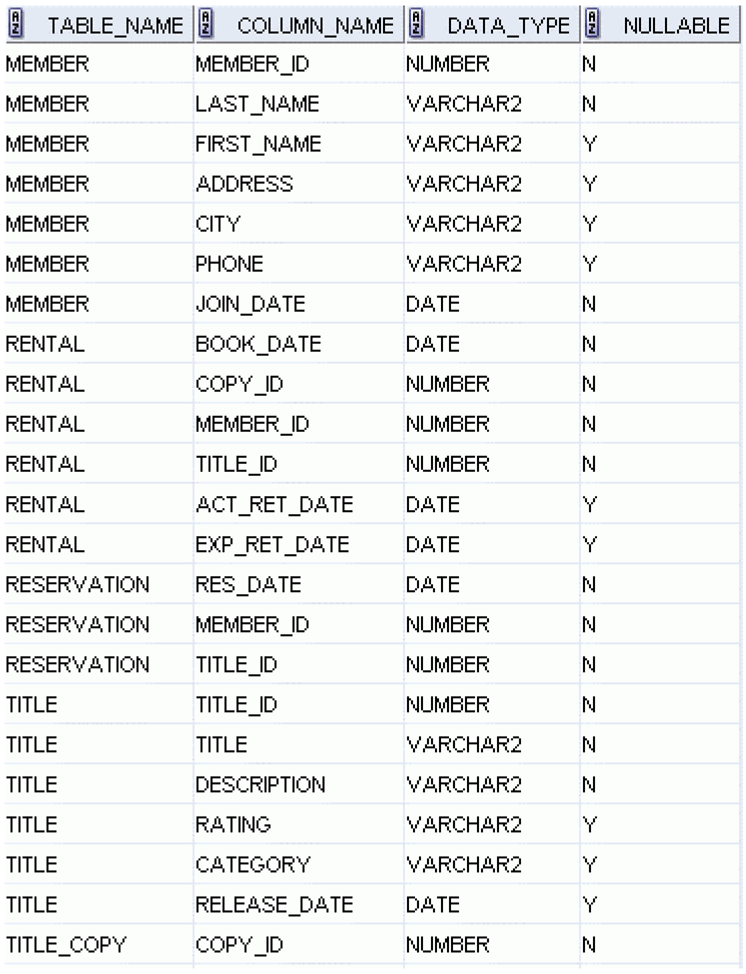
The following is a diagram of the tables and columns that you created for the video application:



Note: First, run the `dropvid.sql` script in the labs folder to drop tables if they already exist. Then run the `buildvid.sql` script in the labs folder to create and populate the tables.

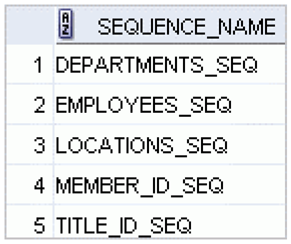
Additional Practices: Case Study (continued)

1. Verify that the tables were created properly by running a report to show the list of tables and their column definitions.



TABLE_NAME	COLUMN_NAME	DATA_TYPE	NULLABLE
MEMBER	MEMBER_ID	NUMBER	N
MEMBER	LAST_NAME	VARCHAR2	N
MEMBER	FIRST_NAME	VARCHAR2	Y
MEMBER	ADDRESS	VARCHAR2	Y
MEMBER	CITY	VARCHAR2	Y
MEMBER	PHONE	VARCHAR2	Y
MEMBER	JOIN_DATE	DATE	N
RENTAL	BOOK_DATE	DATE	N
RENTAL	COPY_ID	NUMBER	N
RENTAL	MEMBER_ID	NUMBER	N
RENTAL	TITLE_ID	NUMBER	N
RENTAL	ACT_RET_DATE	DATE	Y
RENTAL	EXP_RET_DATE	DATE	Y
RESERVATION	RES_DATE	DATE	N
RESERVATION	MEMBER_ID	NUMBER	N
RESERVATION	TITLE_ID	NUMBER	N
TITLE	TITLE_ID	NUMBER	N
TITLE	TITLE	VARCHAR2	N
TITLE	DESCRIPTION	VARCHAR2	N
TITLE	RATING	VARCHAR2	Y
TITLE	CATEGORY	VARCHAR2	Y
TITLE	RELEASE_DATE	DATE	Y
TITLE_COPY	COPY_ID	NUMBER	N

2. Verify the existence of the `MEMBER_ID_SEQ` and `TITLE_ID_SEQ` sequences in the data dictionary.



	SEQUENCE_NAME
1	DEPARTMENTS_SEQ
2	EMPLOYEES_SEQ
3	LOCATIONS_SEQ
4	MEMBER_ID_SEQ
5	TITLE_ID_SEQ

3. You want to create some users who have access only to their own rentals. Create a user called Carmen and grant her the privilege to select from the `RENTAL` table.
Note: Make sure to prefix the username with your database account. For example, if you are the user `oraxx`, create a user called `oraxx_Carmen`.

Additional Practices: Case Study (continued)

4. Add a price column (number 4,2) to the `TITLE` table to store how much it costs to rent the title.

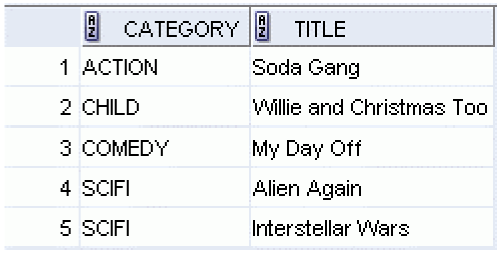
5. Add a `CATEGORY` table to store `CATEGORY_ID` and `CATEGORY_DESCRIPTION`. The table has a foreign key with the `CATEGORY` column in the `TITLE` table.

6. Select all the tables from the data dictionary.

7. There is no real need to store reservations any longer. You can drop the table.

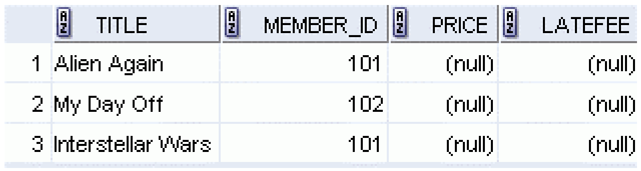
8. Create a `RENTAL_HISTORY` table to store the details of a rental by member for the last six months. (**Hint:** You can copy the `RENTAL` table.)

9. Show a list of the top 10 titles rented in the last month grouped by category.



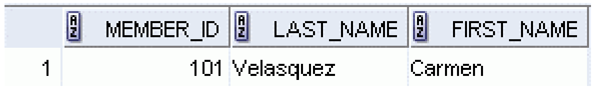
	CATEGORY	TITLE
1	ACTION	Soda Gang
2	CHILD	Willie and Christmas Too
3	COMEDY	My Day Off
4	SCIFI	Alien Again
5	SCIFI	Interstellar Wars

10. You want to calculate the late fee (price of title/day) if the member brings back the video six days late.



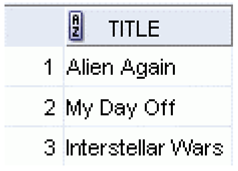
	TITLE	MEMBER_ID	PRICE	LATEFEE
1	Alien Again	101	(null)	(null)
2	My Day Off	102	(null)	(null)
3	Interstellar Wars	101	(null)	(null)

11. Show a list of members who have rented two or more times.



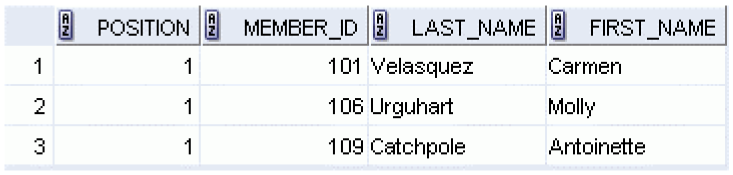
	MEMBER_ID	LAST_NAME	FIRST_NAME
1	101	Velasquez	Carmen

12. Show a list of titles who have a status of rented.



	TITLE
1	Alien Again
2	My Day Off
3	Interstellar Wars

13. Show a list of members who have “99” in their phone numbers.



	POSITION	MEMBER_ID	LAST_NAME	FIRST_NAME
1	1	101	Velasquez	Carmen
2	1	106	Urguhart	Molly
3	1	109	Catchpole	Antoinette

Additional Practice Solutions

The following exercises can be used for extra practice after you have discussed data manipulation language (DML) and data definition language (DDL) statements in the lessons titled "Managing Schema Objects" and "Manipulating Large Data Sets."

Note: Run the `lab_ap_cre_special_sal.sql`, `lab_ap_cre_sal_history.sql`, and `lab_ap_cre_mgr_history.sql` scripts in the labs folder to create the `SPECIAL_SAL`, `SAL_HISTORY`, and `MGR_HISTORY` tables.

1. The Human Resources department wants to get a list of underpaid employees, salary history of employees, and salary history of managers based on an industry salary survey. So they have asked you to do the following:

   Write a statement to do the following:

   - Retrieve details such as 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 details such as the employee ID and salary into the `SPECIAL_SAL` table.
   - Insert details such as the employee ID, hire date, and salary into the `SAL_HISTORY` table.
   - Insert details such as the 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;
```

2. 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;
```

3. Nita, the DBA, needs you to create a table, which has a primary key constraint, but she wants to name the index to have a different name than the constraint. 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	
Data Type	Number	VARCHAR2
Length	4	30

```
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));
```

4. 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';
```

The following exercises can be used for extra practice after you have discussed datetime functions.

You work for a global company and the new vice president of operations wants to know the different time zones of all the company branches. The new vice president has requested the following information:

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

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

6. a. Write queries to display the time zone offsets (`TZ_OFFSET`) for the following time zones:
   - Australia/Sydney

```
SELECT TZ_OFFSET ('Australia/Sydney') from dual;
```

   - Chile/Easter Island

```
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 = '+10:00';
```

   c. Display `SYSDATE`, `CURRENT_DATE`, `CURRENT_TIMESTAMP`, and `LOCALTIMESTAMP` for this session.
   Note: The output may 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/Easter Island.

   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 may differ, based on daylight saving time.

```
ALTER SESSION SET TIME_ZONE = '-06:00';
```

   e. Display `SYSDATE`, `CURRENT_DATE`, `CURRENT_TIMESTAMP`, and `LOCALTIMESTAMP` for this session.

Note: The output may be different based on the date when the command is executed.

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

f. Alter the session to set `NLS_DATE_FORMAT` to `DD-MON-YYYY`.

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MON-YYYY';
```

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.
- 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 may differ, based on daylight saving time.

7. The Human Resources department wants a list of employees who are up for review in January, so they have requested you to do the following:

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

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

The following exercises can be used for extra practice after you have discussed advanced subqueries.

8. The CEO needs a report on the top three earners in the company for profit sharing. You are responsible to provide the CEO with a list.

   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);
```

9. The benefits for the state of California have been changed based on a local ordinance. So the benefits representative has asked you to compile a list of the people who are affected. 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'));
```

10. Nita, the DBA, wants to remove old information from the database. One of the things she thinks is unnecessary is the old employment records. She has asked you to do the following:

    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 two jobs.
    Hint: Use a correlated `DELETE` command.

```
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));
```

11. The vice president of Human Resources needs the complete employment records for the annual employee recognition banquet speech. The vice president makes a quick phone call to stop you from following the DBA's orders.

    Roll back the transaction.

```
ROLLBACK;
```

12. The sluggish economy is forcing management to take cost reduction actions. The CEO wants to review the highest paid jobs in the company. You are responsible to provide the CEO with a list based on the following specifications:

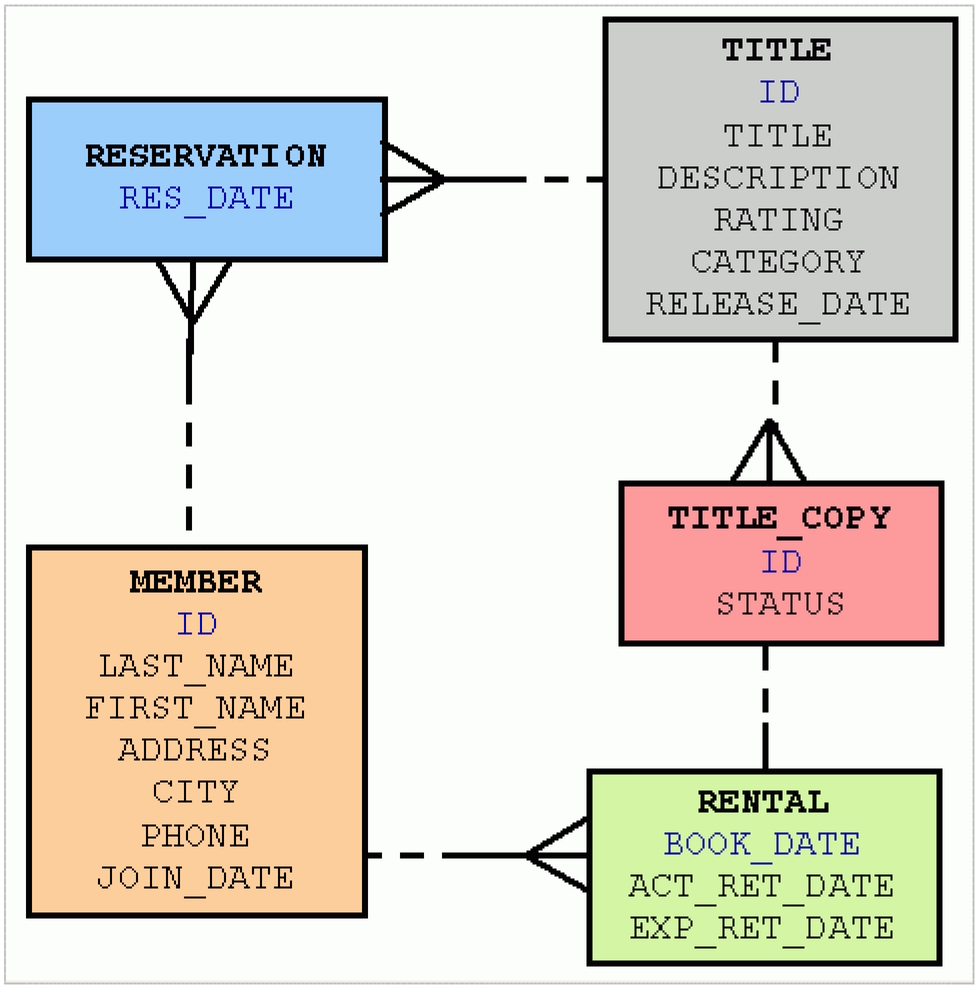
    Write a query to display the job IDs of those jobs whose maximum salary is above half the maximum salary in the entire company. Use the `WITH` clause to write this query. Name the query `MAX_SAL_CALC`.

```
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;
```

Additional Practices: Case Study Solutions

In the case study for the *SQL Fundamentals I* course, you built a set of database tables for a video application. In addition, you inserted, updated, and deleted records in a video store database and generated a report.

The following is a diagram of the tables and columns that you created for the video application:



Note: First, run the `dropvid.sql` script in the labs folder to drop tables if they already exist. Then run the `buildvid.sql` script in the labs folder to create and populate the tables.

Additional Practices: Case Study Solutions (continued)

1. Verify that the tables were created properly by running a report to show the list of tables and their column definitions.

```
SELECT table_name,column_name,data_type,nullable
FROM user_tab_columns
WHERE table_name
IN('MEMBER','TITLE','TITLE_COPY','RENTAL','RESERVATION');
```

2. Verify the existence of the `MEMBER_ID_SEQ` and `TITLE_ID_SEQ` sequences in the data dictionary.

```
SELECT sequence_name FROM user_sequences;
```

3. You want to create some users who have access only to their own rentals. Create a user called Carmen and grant her the privilege to select from the `RENTAL` table.
   Note: Make sure to prefix the username with your database account. For example, if you are the user `oraxx`, create a user called `oraxx_Carmen`.

```
CREATE USER oraxx_carmen IDENTIFIED BY oracle ;
GRANT select ON rental TO oraxx_carmen;
```

4. Add a price column (number 4,2) to the `TITLE` table to store how much it costs to rent the title.

```
ALTER TABLE title ADD(price NUMBER(6))
```

5. Add a `CATEGORY` table to store `CATEGORY_ID` and `CATEGORY_DESCRIPTION`. The table has a foreign key with the `CATEGORY` column in the `TITLE` table.

```
CREATE TABLE CATEGORY
   (  "CATEGORY_ID" NUMBER(6,0) NOT NULL ENABLE,
      "CATEGORY_DESCRIPTION" VARCHAR2(4000 BYTE),
       CONSTRAINT "CATEGORY_PK" PRIMARY KEY ("CATEGORY_ID"))
```

6. Select all the tables from the data dictionary.

```
SELECT table_name FROM user_tables order by table_name;
```

7. There is no real need to store reservations any longer. You can drop the table.

```
DROP TABLE reservation cascade constraints;
```

8. Create a RENTAL_HISTORY table to store the details of a rental by member for the last six months. (**Hint:** You can copy the RENTAL table.)

```
 CREATE TABLE rental_history as select * from rental where '1' = '1'
```

Additional Practices: Case Study Solutions (continued)

9. Show a list of the top 10 titles rented in the last month grouped by category.

```
SELECT t.CATEGORY, t.TITLE
FROM TITLE t, RENTAL r
WHERE t.TITLE_ID = r.TITLE_ID AND
      r. BOOK_DATE > (SYSDATE - 30) AND
      rownum < 10
order by category;
```

10. You want to calculate the late fee (price of title/day) if the member brings back the video six days late.

```
SELECT t.title, m.member_id, t.price, (t.price*6) latefee
FROM title t, member m, rental r
WHERE t.title_id = r.title_id AND
   m.member_id = r.member_id AND
   r.act_ret_date is null;
```

11. Show a list of members who have rented two or more times.

```
SELECT member_id, last_name, first_name FROM member m
where 2 <= (select count(*) from rental_history where member_id =
m.member_id);
```

12. Show a list of titles who have a status of rented.

```
SELECT t.title
FROM title t
JOIN (select title_id, status from title_copy) b
ON t.title_id = b.title_id AND b.status = 'RENTED';
```

13. Show a list of members who have “99” in their phone numbers.

```
SELECT REGEXP_COUNT(phone,'99',1,'i') position, member_id, last_name,
first_name
FROM member
WHERE REGEXP_COUNT(phone,'99',1,'i') > 0;
```