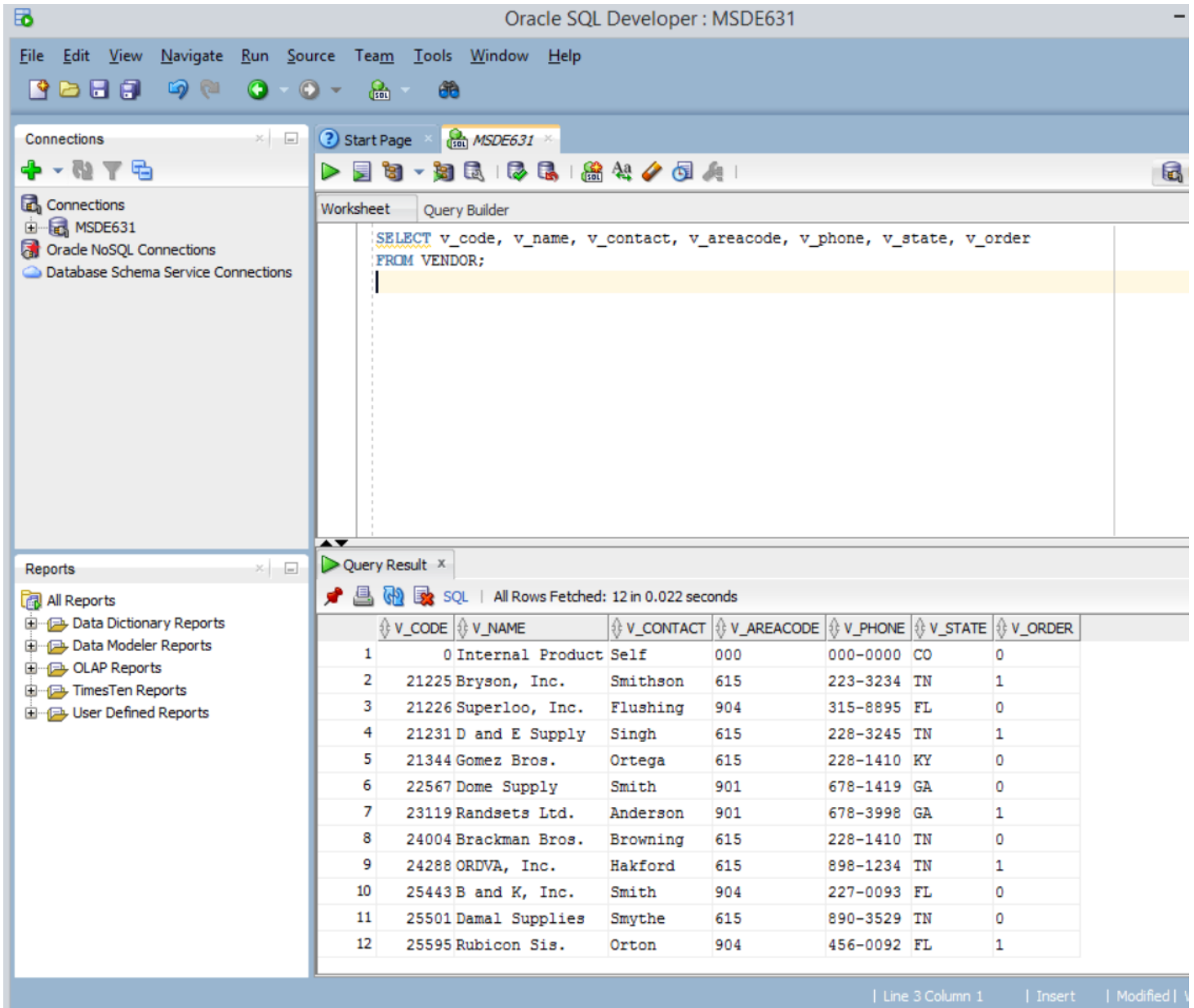


MSDE 631 – Week 1 – Lab

First the information was retrieved from the table **VENDOR**.



Oracle SQL Developer : MSDE631

File Edit View Navigate Run Source Team Tools Window Help

Connections

- Connections
- MSDE631
- Oracle NoSQL Connections
- Database Schema Service Connections

Start Page MSDE631

Worksheet Query Builder

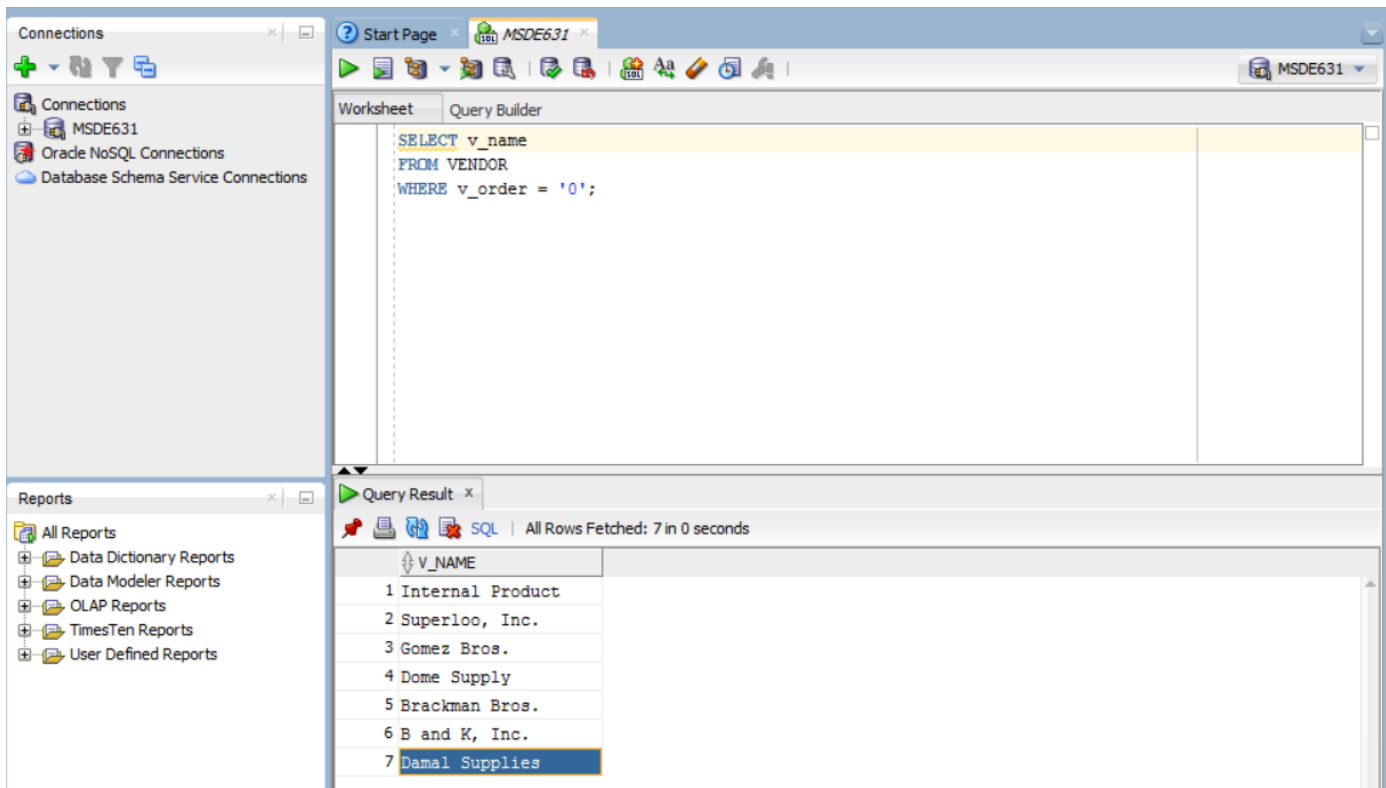
```
SELECT v_code, v_name, v_contact, v_areacode, v_phone, v_state, v_order  
FROM VENDOR;
```

Query Result x

SQL | All Rows Fetched: 12 in 0.022 seconds

	V_CODE	V_NAME	V_CONTACT	V_AREACODE	V_PHONE	V_STATE	V_ORDER
1	0	Internal Product	Self	000	000-0000	CO	0
2	21225	Bryson, Inc.	Smithson	615	223-3234	TN	1
3	21226	Superloo, Inc.	Flushing	904	315-8895	FL	0
4	21231	D and E Supply	Singh	615	228-3245	TN	1
5	21344	Gomez Bros.	Ortega	615	228-1410	KY	0
6	22567	Dome Supply	Smith	901	678-1419	GA	0
7	23119	Randsets Ltd.	Anderson	901	678-3998	GA	1
8	24004	Brackman Bros.	Browning	615	228-1410	TN	0
9	24288	ORDVA, Inc.	Hakford	615	898-1234	TN	1
10	25443	B and K, Inc.	Smith	904	227-0093	FL	0
11	25501	Damal Supplies	Smythe	615	890-3529	TN	0
12	25595	Rubicon Sis.	Orton	904	456-0092	FL	1

| Line 3 Column 1 | Insert | Modified |



The screenshot shows the SQL Developer interface with the MSDE631 database selected. In the Query Builder, the following SQL query is entered:

```
SELECT v_name
FROM VENDOR
WHERE v_order = '0';
```

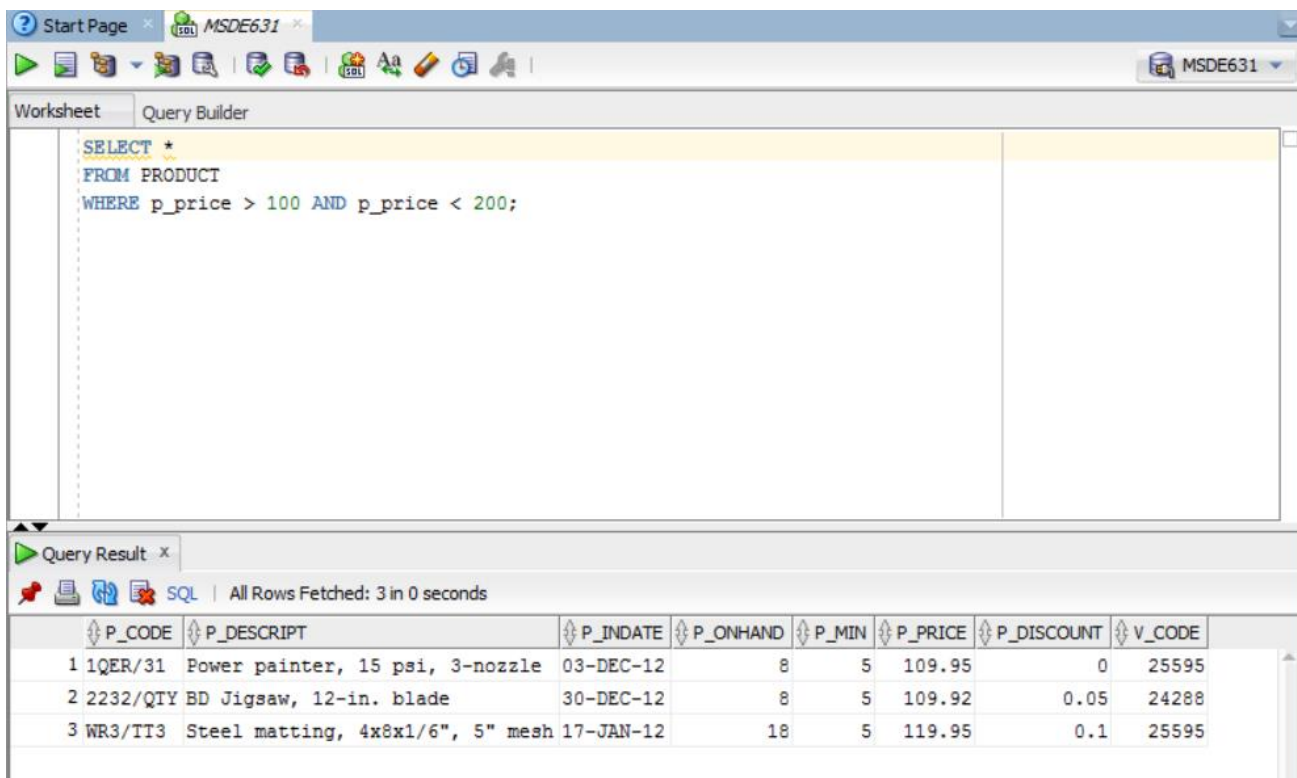
The Query Result window displays the results of the query, showing 7 rows of vendor names:

V_NAME
1 Internal Product
2 Superloo, Inc.
3 Gomez Bros.
4 Dome Supply
5 Brackman Bros.
6 B and K, Inc.
7 Damal Supplies

The above query returned the set of names of vendors from whom we have never ordered.

Using Logical Relationships to Restrict Selected Information:

This query will return an answer set which consists of all the parts with prices between 100 and 200.



The screenshot shows the SQL Developer interface with the MSDE631 database selected. In the Query Builder, the following SQL query is entered:

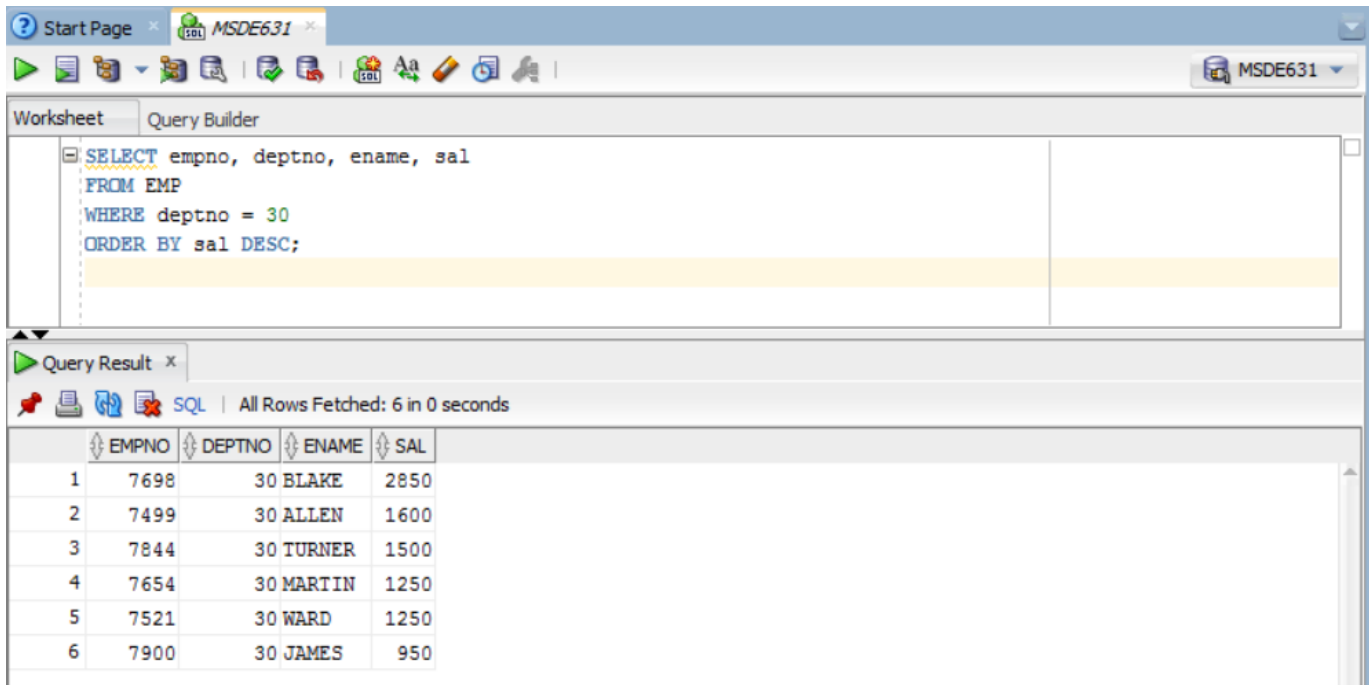
```
SELECT *
FROM PRODUCT
WHERE p_price > 100 AND p_price < 200;
```

The Query Result window displays the results of the query, showing 3 rows of product information:

P_CODE	P_DESCRIPT	P_INDATE	P_ONHAND	P_MIN	P_PRICE	P_DISCOUNT	V_CODE
1 1QER/31	Power painter, 15 psi, 3-nozzle	03-DEC-12	8	5	109.95	0	25595
2 2232/QTY	BD Jigsaw, 12-in. blade	30-DEC-12	8	5	109.92	0.05	24288
3 WR3/TI3	Steel matting, 4x8x1/6", 5" mesh	17-JAN-12	18	5	119.95	0.1	25595

Order/Sort the result set data

The result set of below queries show the employees for department 30 and sorted by the largest monthly salary at the top of the list.



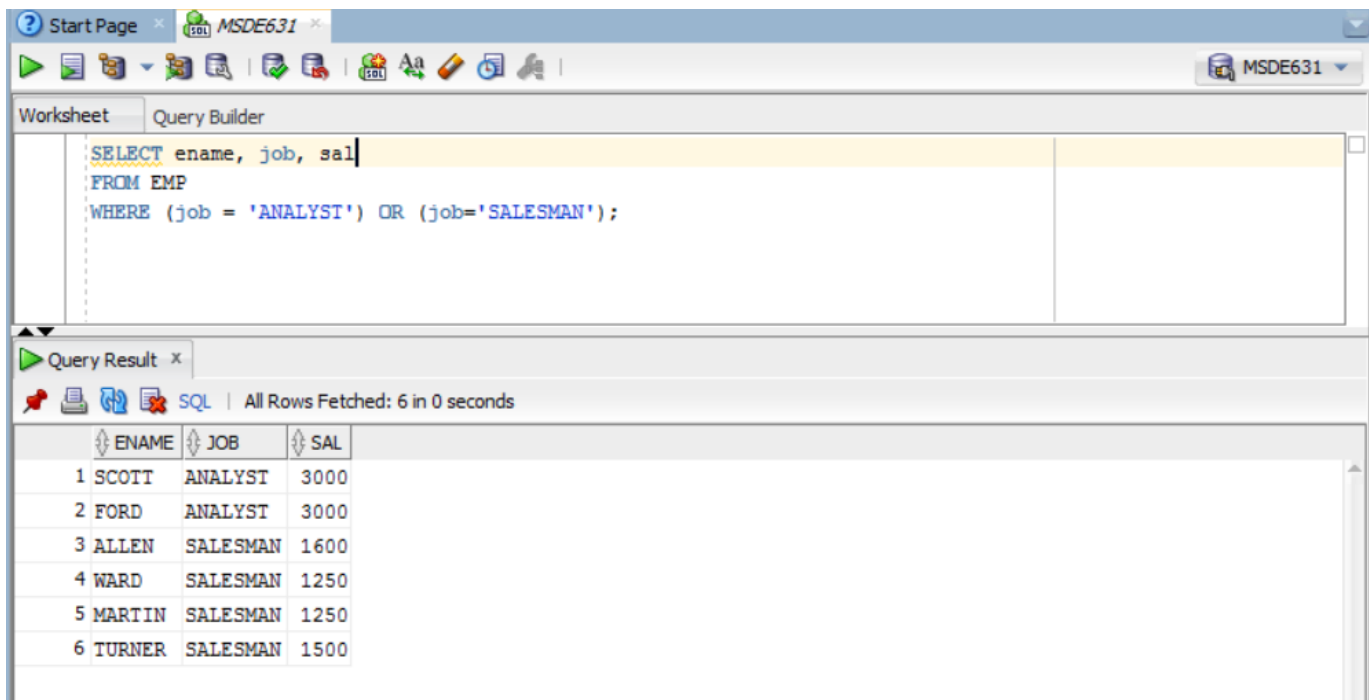
The screenshot shows the MSDE631 Query Builder interface. The SQL query entered is:

```
SELECT empno, deptno, ename, sal  
FROM EMP  
WHERE deptno = 30  
ORDER BY sal DESC;
```

The Query Result pane shows 6 rows fetched in 0 seconds. The results are as follows:

	EMPNO	DEPTNO	ENAME	SAL
1	7698	30	BLAKE	2850
2	7499	30	ALLEN	1600
3	7844	30	TURNER	1500
4	7654	30	MARTIN	1250
5	7521	30	WARD	1250
6	7900	30	JAMES	950

The next command would return a set of employees hired as “analysts” or “salesman”.



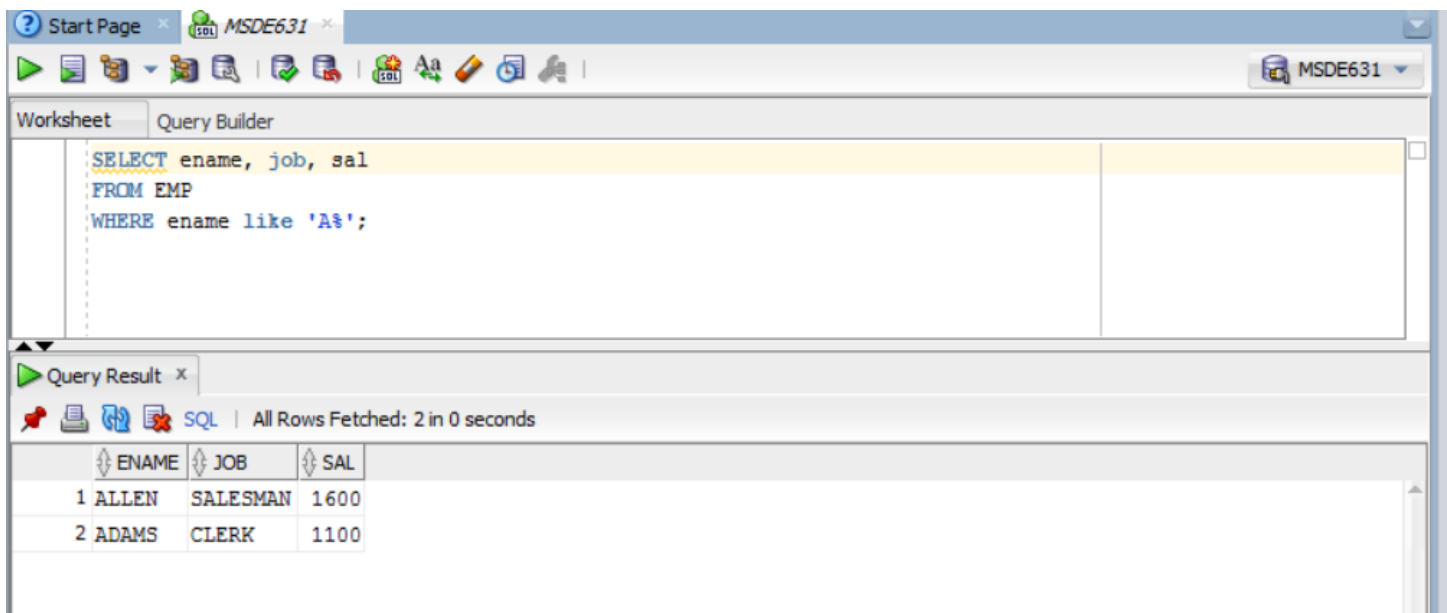
The screenshot shows the MSDE631 Query Builder interface. The SQL query entered is:

```
SELECT ename, job, sal  
FROM EMP  
WHERE (job = 'ANALYST') OR (job='SALESMAN');
```

The Query Result pane shows 6 rows fetched in 0 seconds. The results are as follows:

	ENAME	JOB	SAL
1	SCOTT	ANALYST	3000
2	FORD	ANALYST	3000
3	ALLEN	SALESMAN	1600
4	WARD	SALESMAN	1250
5	MARTIN	SALESMAN	1250
6	TURNER	SALESMAN	1500

To query all employees whose names begin with ‘A’, you would use the “%” wildcard as in:



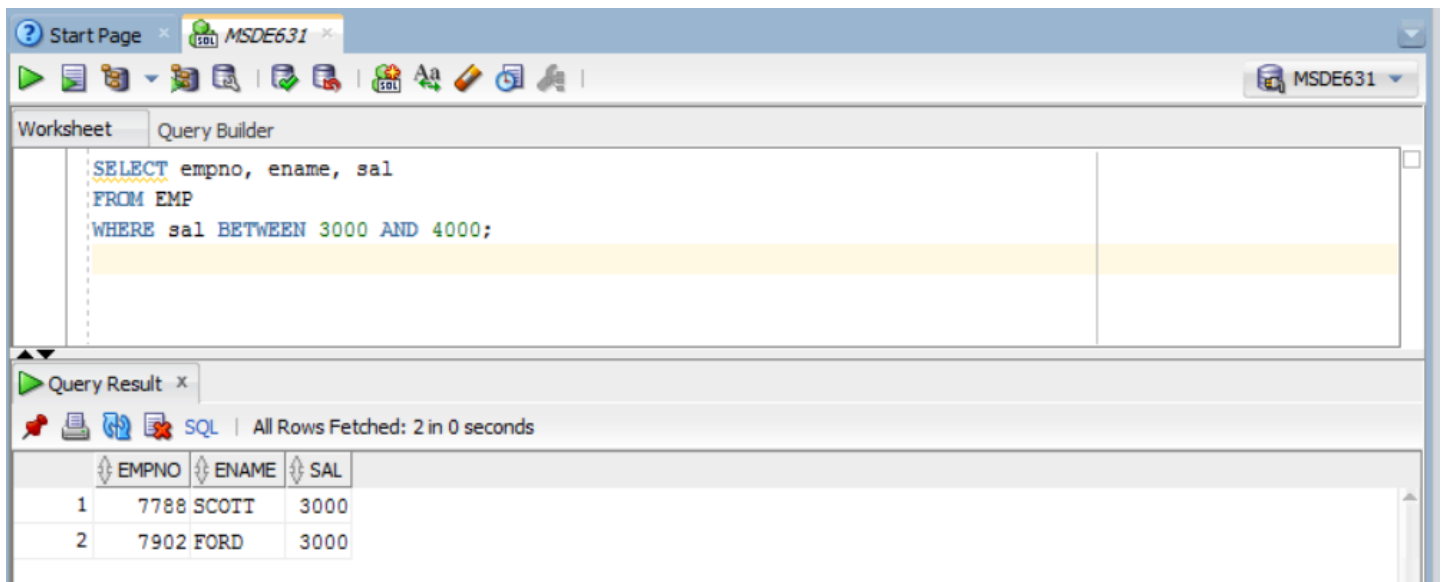
The screenshot shows the SQL Server Enterprise Manager interface. The 'Query Builder' pane contains the following SQL query:

```
SELECT ename, job, sal
FROM EMP
WHERE ename like 'A%';
```

The 'Query Result' pane shows the results of the query, with the text 'All Rows Fetched: 2 in 0 seconds'. The results are displayed in a table with three columns: ENAME, JOB, and SAL.

	ENAME	JOB	SAL
1	ALLEN	SALESMAN	1600
2	ADAMS	CLERK	1100

The following query returns employee information where the salary is greater than or equal to 3000 and less than or equal to 4000:



The screenshot shows the SQL Server Enterprise Manager interface. The 'Query Builder' pane contains the following SQL query:

```
SELECT empno, ename, sal
FROM EMP
WHERE sal BETWEEN 3000 AND 4000;
```

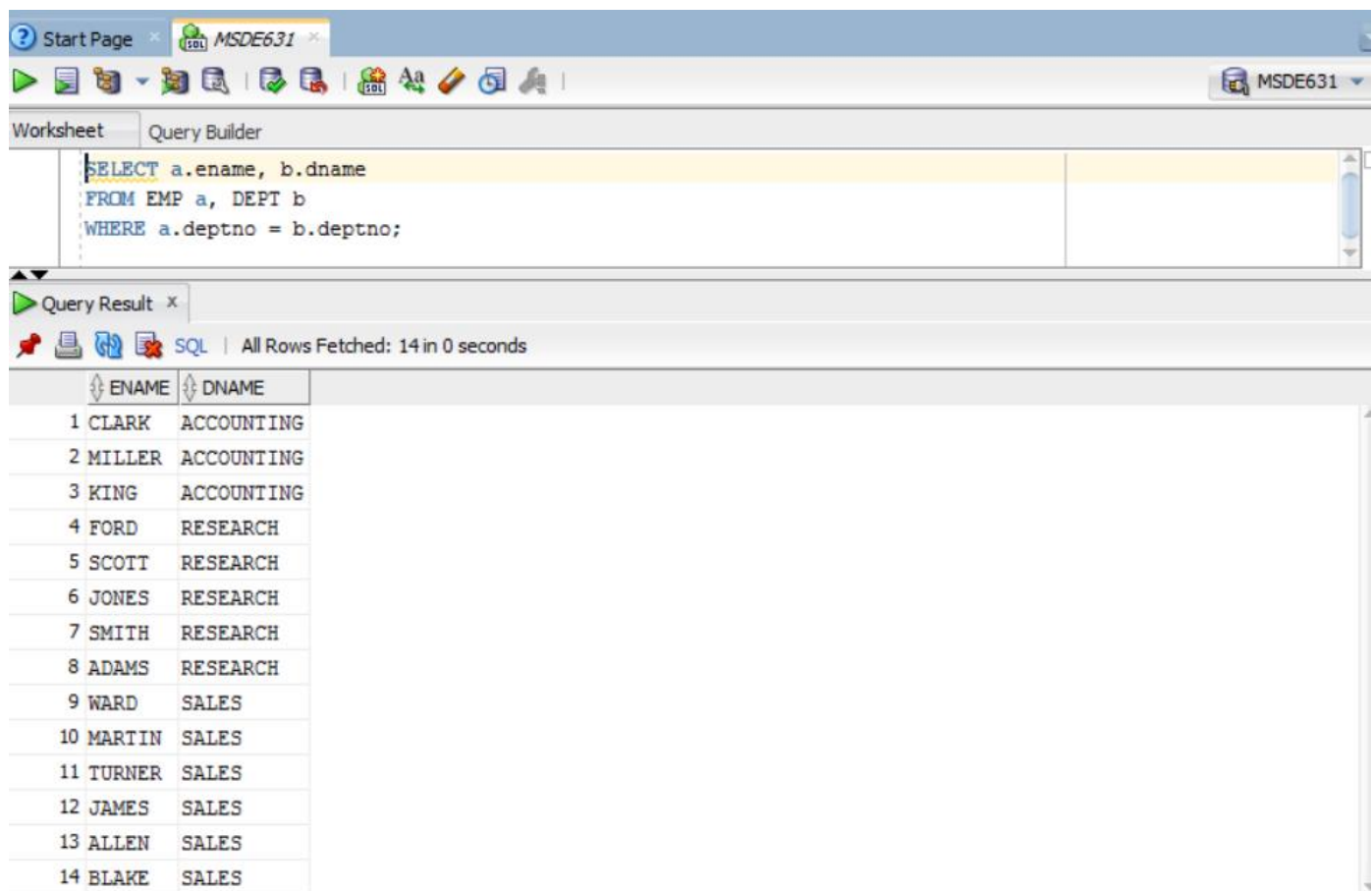
The 'Query Result' pane shows the results of the query, with the text 'All Rows Fetched: 2 in 0 seconds'. The results are displayed in a table with three columns: EMPNO, ENAME, and SAL.

	EMPNO	ENAME	SAL
1	7788	SCOTT	3000
2	7902	FORD	3000

Querying Data from Multiple Tables

Here we perform joint of two tables returning employees names and departments they work for.

Alla Topp
MSDE 631 SQL/NoSQL
Lab 1



The screenshot shows the SQL Server Enterprise Manager interface. The 'Query Builder' tab is active, displaying the following SQL query:

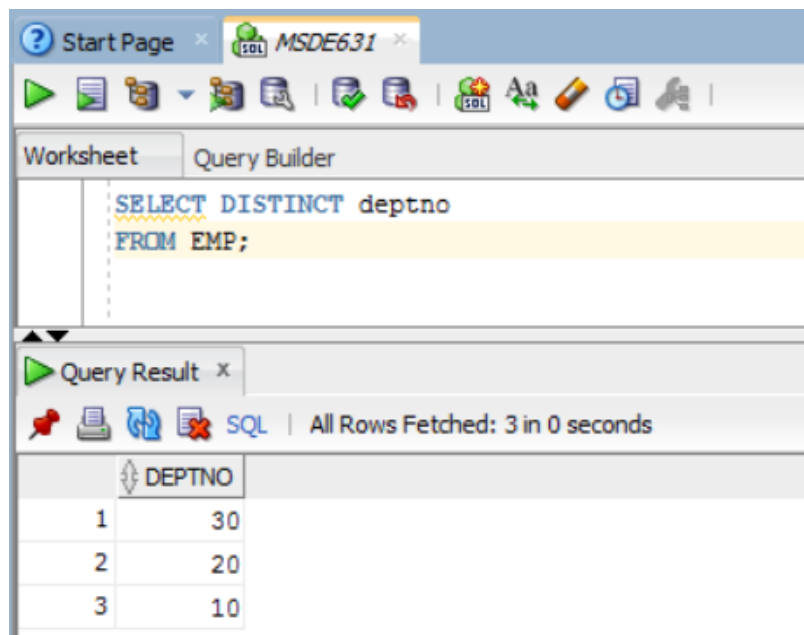
```
SELECT a.ename, b.dname
FROM EMP a, DEPT b
WHERE a.deptno = b.deptno;
```

The 'Query Result' pane below shows the results of the query, which are 14 rows of employee data. The status bar indicates 'All Rows Fetched: 14 in 0 seconds'.

	ENAME	DNAME
1	CLARK	ACCOUNTING
2	MILLER	ACCOUNTING
3	KING	ACCOUNTING
4	FORD	RESEARCH
5	SCOTT	RESEARCH
6	JONES	RESEARCH
7	SMITH	RESEARCH
8	ADAMS	RESEARCH
9	WARD	SALES
10	MARTIN	SALES
11	TURNER	SALES
12	JAMES	SALES
13	ALLEN	SALES
14	BLAKE	SALES

Querying Unique Data

Next query returns a distinct set of department numbers from the EMP table.



The screenshot shows the SQL Server Enterprise Manager interface. The 'Query Builder' tab is active, displaying the following SQL query:

```
SELECT DISTINCT deptno
FROM EMP;
```

The 'Query Result' pane below shows the results of the query, which are 3 rows of distinct department numbers. The status bar indicates 'All Rows Fetched: 3 in 0 seconds'.

	DEPTNO
1	30
2	20
3	10

Aggregate Functions and Scalar functions

Below query would return the average salary from the EMP table.

The screenshot shows the SQL Server Enterprise Manager interface. The 'Query Builder' tab is active, displaying the following SQL query:

```
SELECT ROUND (AVG (sal) , 2)
FROM EMP;
```

Below the query, the 'Query Result' pane shows the results of the query. The results are displayed in a table with one row and one column:

ROUND(AVG(SAL),2)
2073.21

This would return the number of records in the EMP table.

The screenshot shows the SQL Server Enterprise Manager interface. The 'Query Builder' tab is active, displaying the following SQL query:

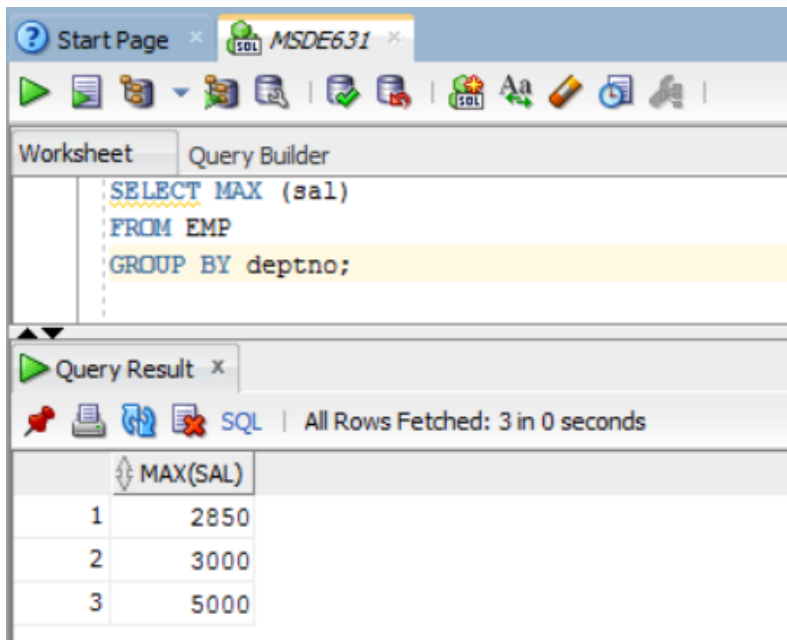
```
SELECT COUNT (*)
FROM EMP;
```

Below the query, the 'Query Result' pane shows the results of the query. The results are displayed in a table with one row and one column:

COUNT(*)
14

Next would return the maximum salary within (GROUP BY) each department in the EMP table.

Alla Topp
MSDE 631 SQL/NoSQL
Lab 1



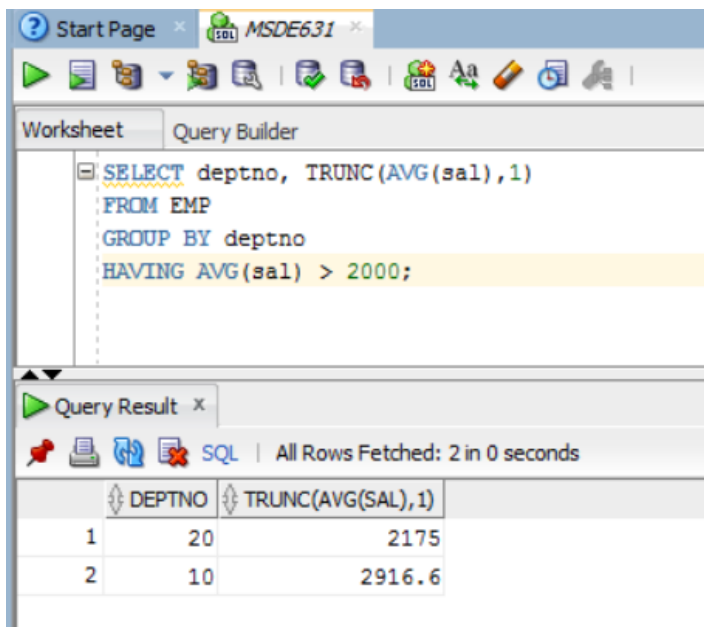
The screenshot shows the SQL Server Enterprise Manager interface. The 'Query Builder' tab is active, displaying the following SQL query:

```
SELECT MAX (sal)
FROM EMP
GROUP BY deptno;
```

Below the query, the 'Query Result' tab shows the results of the query. The status bar indicates 'All Rows Fetched: 3 in 0 seconds'. The results are displayed in a table with two columns: 'MAX(SAL)' and 'DEPTNO'.

MAX(SAL)	DEPTNO
2850	1
3000	2
5000	3

The HAVING clause further filters the data by the given criteria (here we only display records where the average department salary is more than 2,000).



The screenshot shows the SQL Server Enterprise Manager interface. The 'Query Builder' tab is active, displaying the following SQL query:

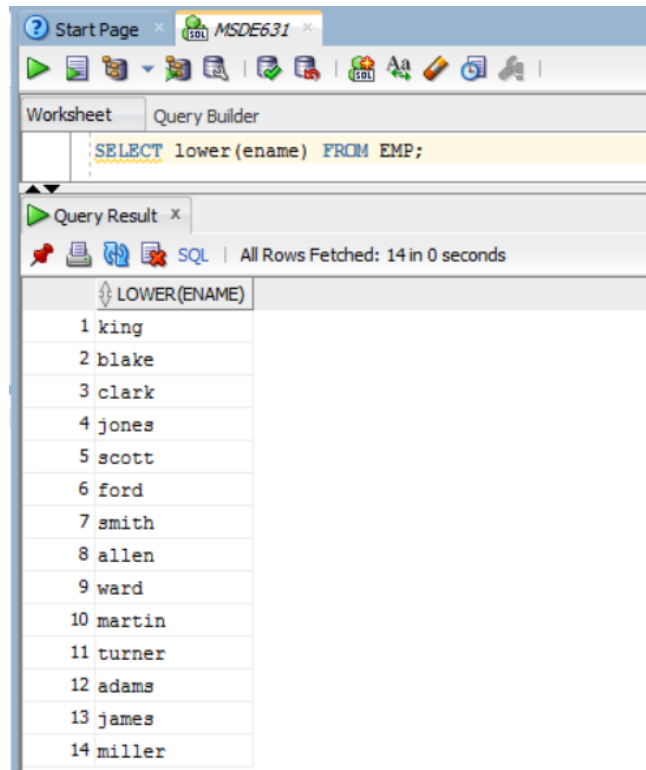
```
SELECT deptno, TRUNC (AVG(sal),1)
FROM EMP
GROUP BY deptno
HAVING AVG(sal) > 2000;
```

Below the query, the 'Query Result' tab shows the results of the query. The status bar indicates 'All Rows Fetched: 2 in 0 seconds'. The results are displayed in a table with two columns: 'DEPTNO' and 'TRUNC(AVG(SAL),1)'.

DEPTNO	TRUNC(AVG(SAL),1)
20	2175
10	2916.6

Alla Topp
MSDE 631 SQL/NoSQL
Lab 1

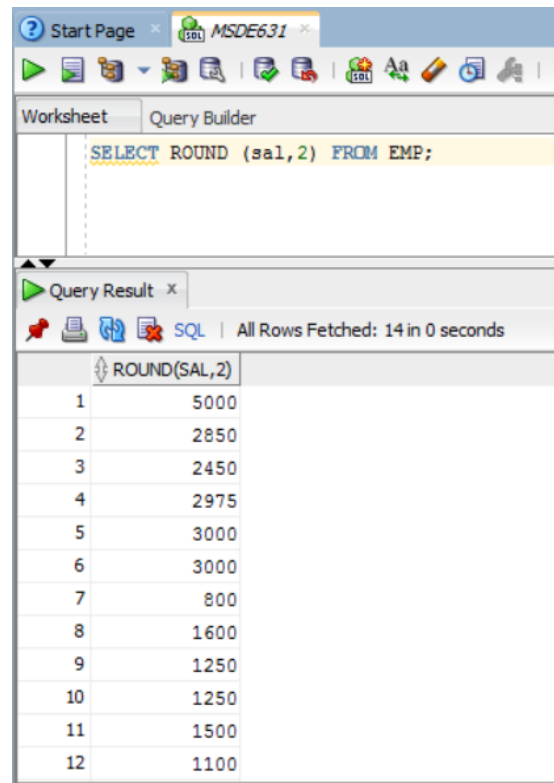
Next query returns employee names in lower case.



The screenshot shows the MSDE631 Query Builder interface. The SQL query entered is `SELECT lower(ename) FROM EMP;`. The Query Result pane shows 14 rows fetched in 0 seconds. The results are displayed in a table with the column header `LOWER(ENAME)`.

	LOWER(ENAME)
1	king
2	blake
3	clark
4	jones
5	scott
6	ford
7	smith
8	allen
9	ward
10	martin
11	turner
12	adams
13	james
14	milller

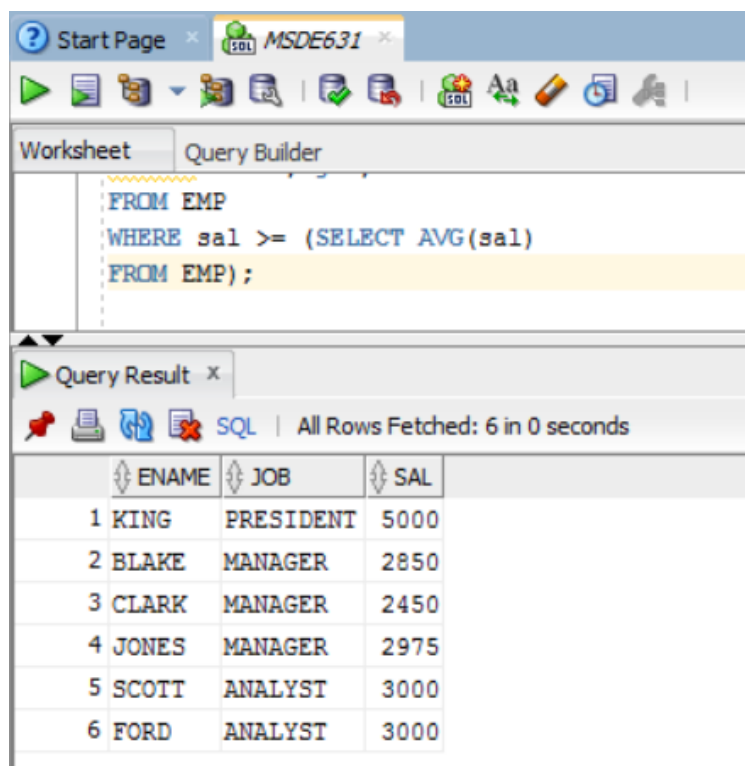
Below returns the salary column rounded to 2



The screenshot shows the MSDE631 Query Builder interface. The SQL query entered is `SELECT ROUND (sal,2) FROM EMP;`. The Query Result pane shows 14 rows fetched in 0 seconds. The results are displayed in a table with the column header `ROUND(SAL,2)`.

	ROUND(SAL,2)
1	5000
2	2850
3	2450
4	2975
5	3000
6	3000
7	800
8	1600
9	1250
10	1250
11	1500
12	1100

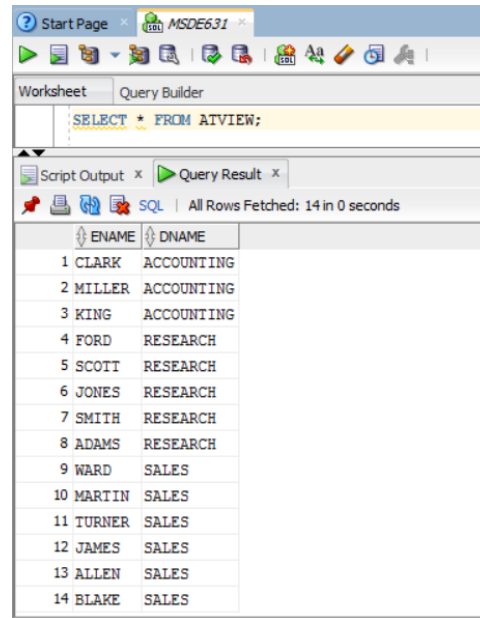
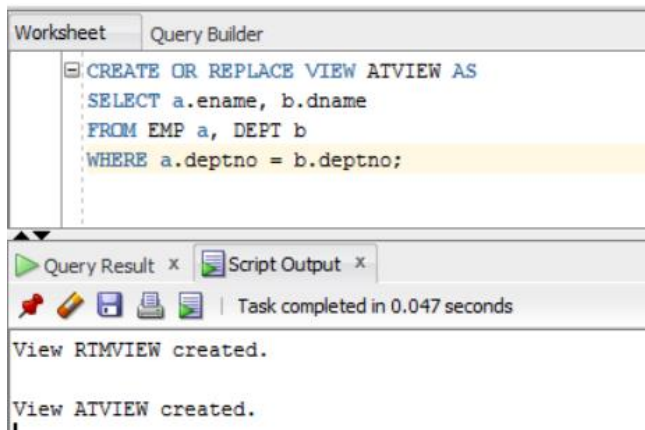
Next query returns the names, job title and salary of people making more than the average salary:



The screenshot shows the MSDE631 Query Builder interface. The SQL query entered is `FROM EMP WHERE sal >= (SELECT AVG(sal) FROM EMP);`. The Query Result pane shows 6 rows fetched in 0 seconds. The results are displayed in a table with the column headers `ENAME`, `JOB`, and `SAL`.

	ENAME	JOB	SAL
1	KING	PRESIDENT	5000
2	BLAKE	MANAGER	2850
3	CLARK	MANAGER	2450
4	JONES	MANAGER	2975
5	SCOTT	ANALYST	3000
6	FORD	ANALYST	3000

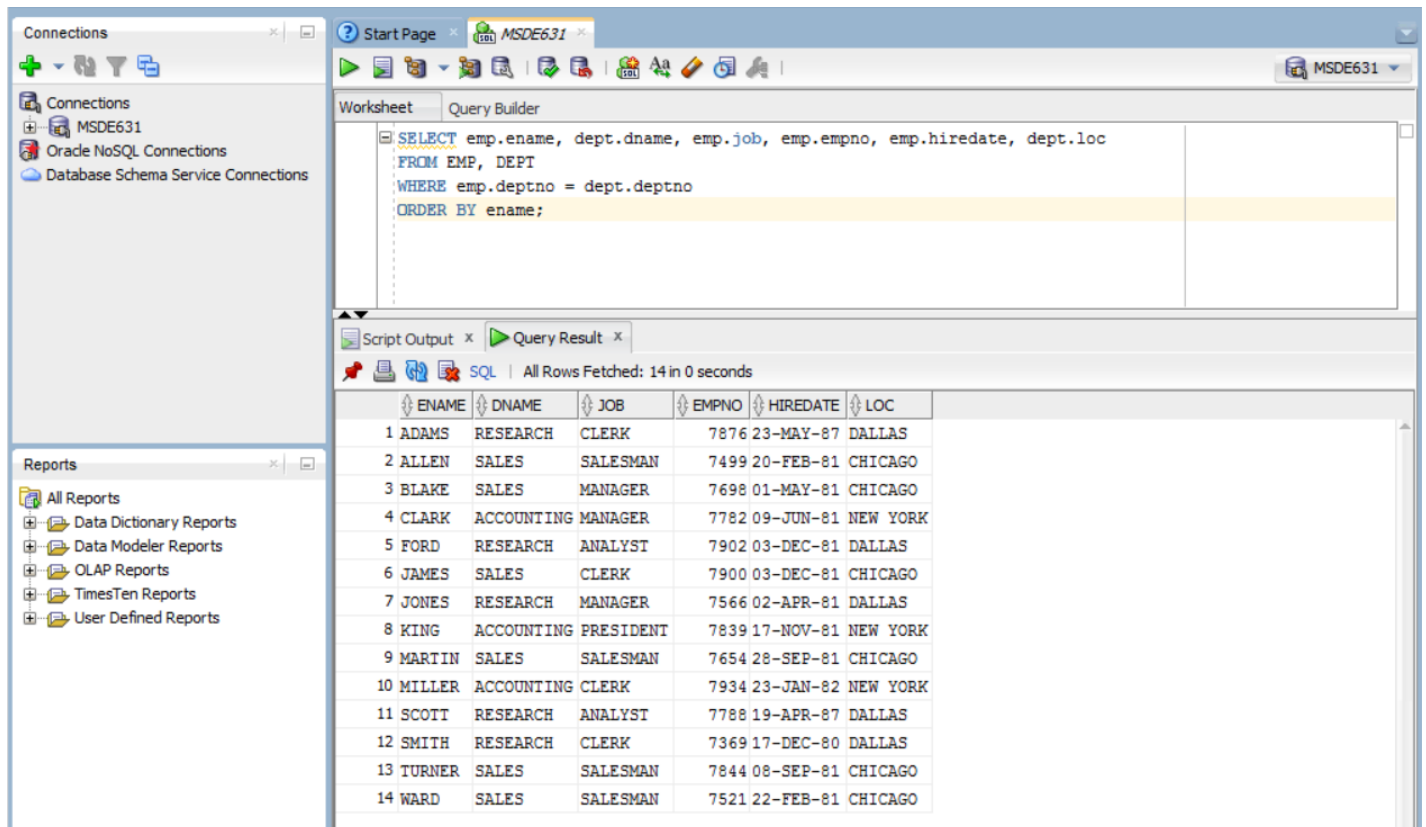
Views



Challenge:

Write two queries on your own using the tables found in the MSDE631 schema.

- 1) The first query will join the emp and dept tables together. The columns to include are shown below. Sort the results by ename. Your result set should match the results shown below exactly:



- 2) For the second query, you might have to Google Oracle SQL commands to write the SQL to match the result set shown below. You will join the emp and dept tables together. The columns to include are shown below, notice that I changed the column names using column aliases. Sort the results descending order by the column named count_of_employees. Your result set should match the results shown below exactly:

The screenshot shows the Oracle SQL Developer interface for the MSDE631 database. The left pane displays the database schema with tables DEPT and EMP. The DEPT table has columns DEPTNO, DNAME, and LOC. The EMP table has columns EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, and DEPTNO. The right pane shows a SQL query in the Worksheet:

```
SELECT dname, COUNT(*) count_of_employees, ROUND(AVG(sal),2) as average_salary
FROM dept, emp
WHERE dept.deptno = emp.deptno
GROUP BY DNAME
ORDER BY 2 DESC;
```

The bottom pane shows the Query Result with 3 rows fetched in 0 seconds. The results are as follows:

	DNAME	COUNT_OF_EMPLOYEES	AVERAGE_SALARY
1	SALES	6	1566.67
2	RESEARCH	5	2175
3	ACCOUNTING	3	2916.67