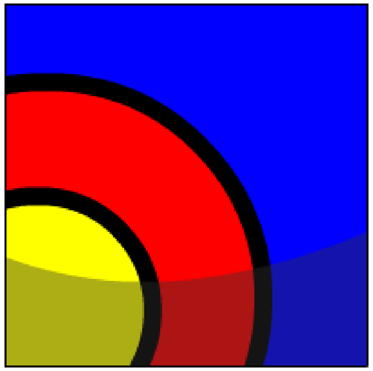
Final Exam Review

What Will I Learn?



In this lesson, you will learn to:

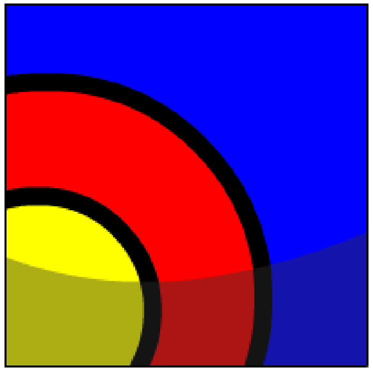
- Review the key points about Case and Character Manipulation
- Review Number, Date, Conversion and General Functions
- Review conditional expressions
- Review Cartesian Product and Join Operations
- Review Non-equijoins, outer joins, self joins, cross joins, natural joins and join clauses
- Review group functions, group by syntax and having clauses
- Review single-row and multiple row subqueries

ORACLE Academy

What Will I Learn?

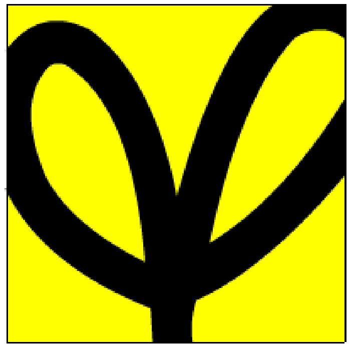
In this lesson, you will learn to:

- Review inserting, updating, and deleting data
- Review default values and the merge statement
- Review creating tables, specifying data types, and modifying a table
- Review not null and unique constraints
- Review primary key, foreign key and check constraints
- Review creating and managing views
- Review creating sequences, indexes and synonyms
- Review creating and revoking object privileges



Why Learn It?

Review is the best preparation for assessment. Assessment allows you to realize how much you've learned and areas you may wish to improve.

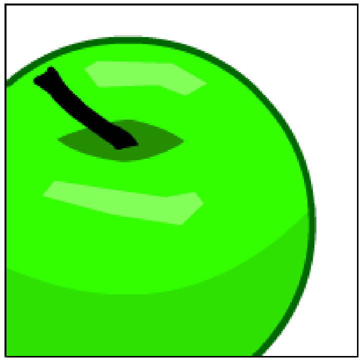


Reviewing the topics learned to this point will help you be your best during the final exam.

Tell Me / Show Me

This is a review of the syntax.

Ensure that you also review the rules concerning the syntax.



Tell Me / Show Me

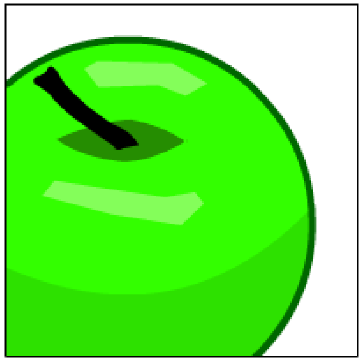
Case and Character Manipulation

Case

LOWER(column name|expression)

UPPER(column name|expression)

INITCAP(column name|expression)



Character

CONCAT(column name|expression, column name|expression)

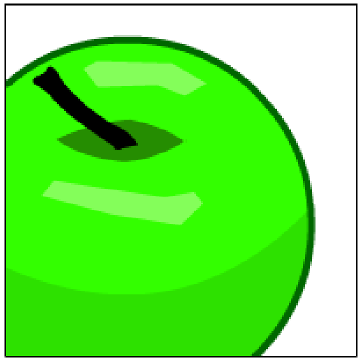
SUBSTR(column name|expression,n,m)

LENGTH(column name|expression)

Tell Me / Show Me

Case and Character Manipulation

Character (cont'd)



INSTR(column name|expression, string literal)

LPAD (column name|expression, n, character literal)

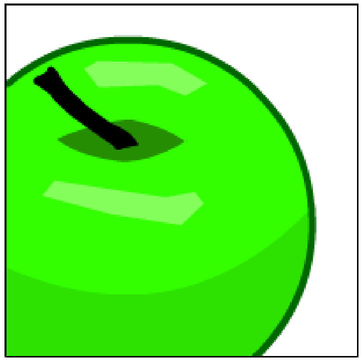
RPAD(column name|expression, n, character literal)

TRIM ([leading | trailing | both] char1 FROM char2)

REPLACE (column name|expression, string to be replaced, replacement string)

Tell Me / Show Me

Number Functions



ROUND(column|expression,n)

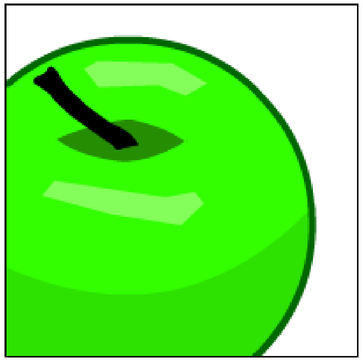
TRUNC(column|expression,n)

MOD(column|expression, column|expression)

ORACLE Academy

Tell Me / Show Me

Date Functions



ROUND(column|expression,string)

TRUNC(column|expression,string)

MONTHS_BETWEEN(column|expression, column|expression)

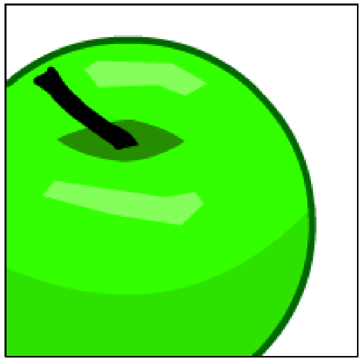
ADD_MONTHS(column|expression,n)

NEXT_DAY(column|expression,’day’)

LAST_DAY(column|expression)

Tell Me / Show Me

Conversion Functions



TO_CHAR(number, ‘format model’)

TO_CHAR(date, ‘format model’)

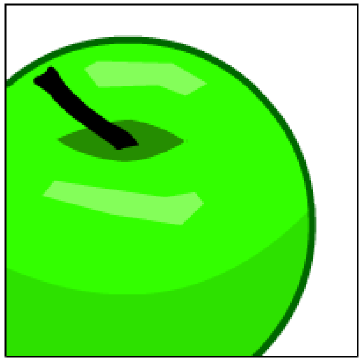
TO_NUMBER(character string, ‘format model’)

TO_DATE(character string, ‘format model’)

$$\begin{tikzcd}
\text{NUMBER} \arrow[r, bend right, "\text{TO\_CHAR}"'] & \text{CHARACTER} \arrow[l, bend right, "\text{TO\_NUMBER}"'] \arrow[r, bend left, "\text{TO\_DATE}"] & \text{DATE} \arrow[l, bend left, "\text{TO\_CHAR}"]
\end{tikzcd}$$

Tell Me / Show Me

NULL Functions



NVL(column|expression, value)

NVL2(column|expression, column|expression, column|expression)

NULLIF(column|expression, column|expression)

COALESCE(column|expression, column|expression, column|expression…. column|expression)

Tell Me / Show Me

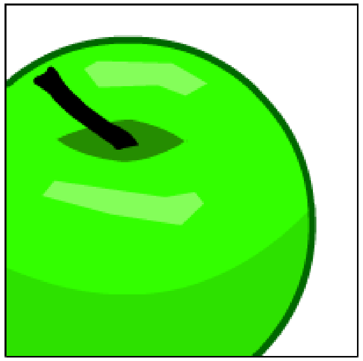
Conditional Expressions

Oracle specific

```
DECODE(column|expression, search1, result1
        [, search2, result2,...,]
        [, default])
```

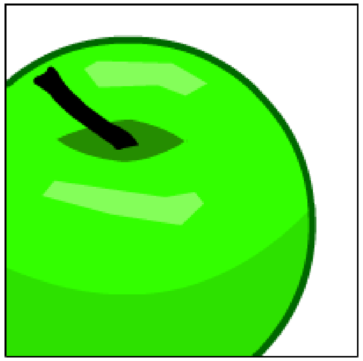
ANSI

```
CASE expr WHEN comparison_expr1 THEN
return_expr1
        [WHEN comparison_expr2 THEN return_expr2
         WHEN comparison_exprn THEN return_exprn
         ELSE else_expr]
END
```



Tell Me / Show Me

Cartesian Product and Join Operations



Cartesian Product

SELECT last_name, department_name

FROM employees, departments;

Oracle Proprietary Joins (equivalent ANSI joins given in parenthesis)

Equijoin (Natural Join, Join .. Using, Join .. On)

SELECT e.employee_id, e.last_name, e.department_id, d.department_name

FROM employees e, departments d

WHERE e.department_id = d.department_id;

Tell Me / Show Me

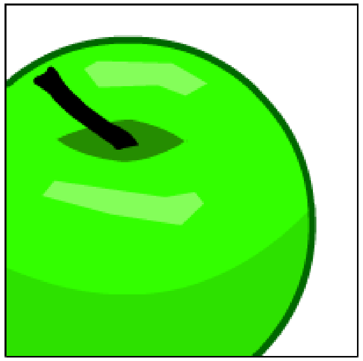
Non-equijoins, Outer Joins

Non-equijoin (Join .. On)

```
SELECT e.employee_id, e.last_name, e.salary,
j.grade_level
FROM employees e, job_grades j
WHERE e.salary >= j.lowest_sal
AND e.salary <= j.highest_sal;
```

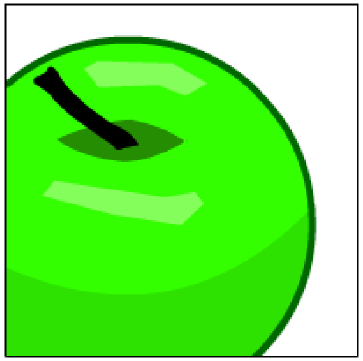
Outer Joins (Right Outer Join, Left Outer Join)

```
SELECT e.employee_id, e.last_name,
e.department_id, d.department_name
FROM employees e, departments d
WHERE e.department_id (+) = d.department_id;
```



Tell Me / Show Me

Non-equijoins, Outer Joins



```
SELECT e.employee_id, e.last_name,
e.department_id, d.department_name
FROM employees e, departments d
WHERE e.department_id = d.department_id(+);
```

Self-Joins (Join .. On)

```
SELECT e.employee_id, e.last_name,
m.employee_id, m.last_name
FROM employees e, employees m
WHERE e.manager_id = m.employee_id;
```

Tell Me / Show Me

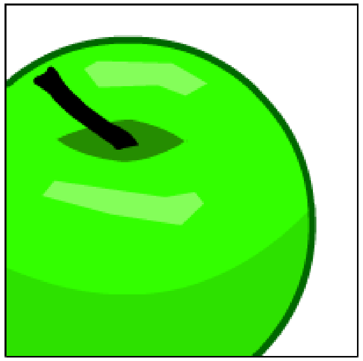
ANSI SQL Standard Syntax (equivalent Oracle specific joins given in parenthesis)

Cross Join (Cartesian Product)

```
SELECT last_name, department_name
FROM employees CROSS JOIN departments;
```

Natural Join (Equijoin)

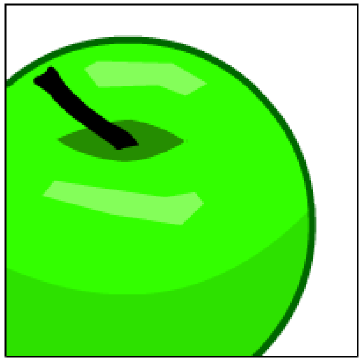
```
SELECT employee_id, last_name,
department_name
FROM employees NATURAL JOIN departments;
```



Tell Me / Show Me

ANSI SQL Standard Syntax (equivalent Oracle specific joins given in parenthesis)

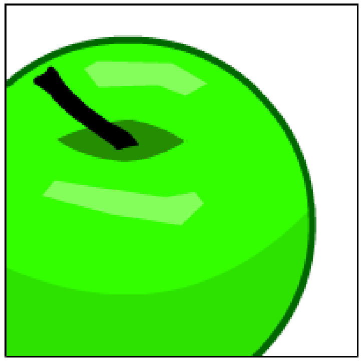
Join .. On (Non equijoin)



```
SELECT e.employee_id, e.last_name, e.salary, j.grade_level
FROM employees e JOIN job_grades j
ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);
```

Tell Me / Show Me

ANSI SQL Standard Syntax (equivalent Oracle specific joins given in parenthesis)



Joins .. Using (Equijoin)

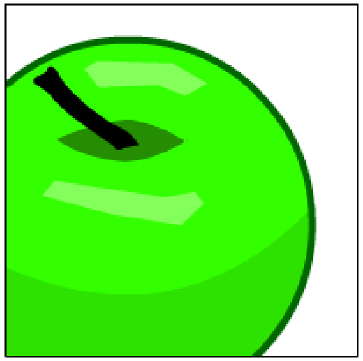
```
SELECT employee_id, last_name, department_name
FROM employees JOIN departments
USING (department_id);
```

Join .. On

```
SELECT e.employee_id, e.last_name, d.department_id,
d.location_id
FROM employees e JOIN departments d
ON (e.department_id = d.department_id);
```

Tell Me / Show Me

ANSI SQL Standard Syntax (equivalent Oracle specific joins given in parenthesis)

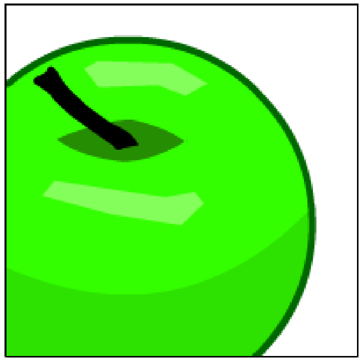


Outer Joins (+)
Right Outer Join
```
SELECT e.employee_id, e.last_name,
e.department_id, d.department_name
FROM employees e RIGHT OUTER JOIN departments d
ON (e.department_id = d.department_id);
```

Left Outer Join
```
SELECT e.employee_id, e.last_name,
e.department_id, d.department_name
FROM employees e LEFT OUTER JOIN departments d
ON (e.department_id = d.department_id);
```

Tell Me / Show Me

ANSI SQL Standard Syntax (equivalent Oracle specific joins given in parenthesis)



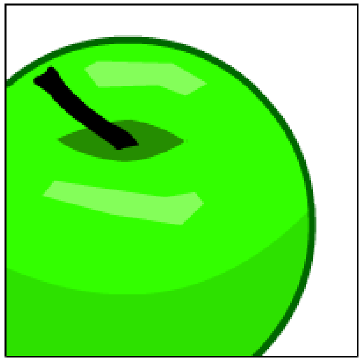
Outer Joins (+)

Full Outer Join (No comparable Oracle specific Join)

```
SELECT e.employee_id, e.last_name,
e.department_id, d.department_name
FROM employees e FULL OUTER JOIN departments d
ON (e.department_id = d.department_id);
```

Tell Me / Show Me

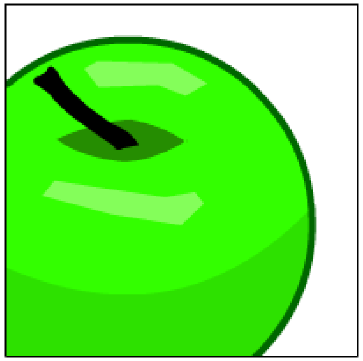
Group Functions, Group By Syntax and Having Clauses



```
AVG (column |expression)
COUNT (column |expression)
MIN (column |expression)
MAX (column |expression)
SUM (column |expression)
VARIANCE (column |expression)
STDDEV (column |expression)

SELECT column1, AVG (column |expression)
FROM table 1
GROUP BY (ROLLUP | CUBE)  (column1 | GROUPING SETS)
HAVING AVG (column |expression)
```

Tell Me / Show Me

Single-row and multiple row Subqueries



```
SELECT column1..
FROM table 1
WHERE column2 =
                    (SELECT column2
                      FROM table 1
                      WHERE column 3 = expression)
```

Single row operators: =,>,<,>=,<=,<>

Multiple row operators: IN, ANY, ALL

Tell Me / Show Me

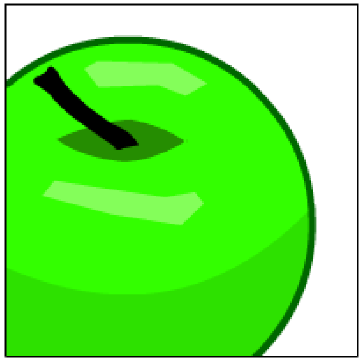
Pairwise and non-pairwise Subqueries

Pairwise

```
SELECT column1..
FROM table 1
WHERE (column2, column3) = (SELECT column2, column3
                            FROM table 1
                           WHERE column 4 = expression)
```

Non-pairwise

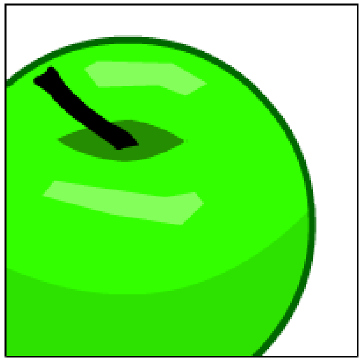
```
SELECT column1..
FROM table 1
WHERE column2 = (SELECT column2
                 FROM table 1
                 WHERE column 4 = expression)
AND       column3 = (SELECT column3
                  FROM table 2
                  WHERE column 4 = expression)
```



Tell Me / Show Me

Correlated Subqueries

```
SELECT o.column1..
FROM table_1 o
WHERE o.column2 =
                    (SELECT i.column2
                      FROM table_2 i
                      WHERE i.column1 = o.column1)
```



Tell Me / Show Me

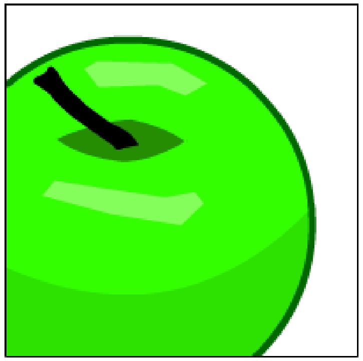
Inserting, Updating and Deleting Data

Explicit insert

```
INSERT INTO table (column1, column2…)
VALUES (value1, value2…) ;
```

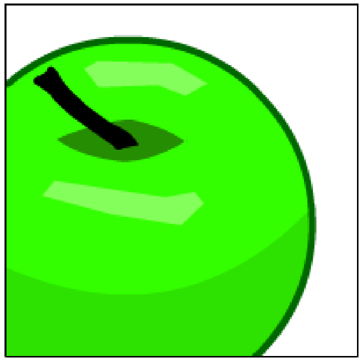
Implicit insert

```
INSERT INTO table
VALUES (value1, value2, value3, value4);

UPDATE table1
SET column1 = value1,
        column2 = value2…
WHERE column1 = value;

DELETE FROM table1
WHERE column1 = value;
```



Tell Me / Show Me

Inserting, Updating and Deleting Data
Multi-table Insert

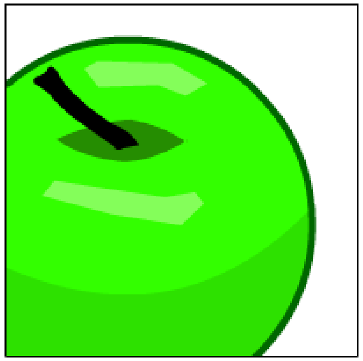


```
conditional_insert_clause
[ ALL | FIRST ]
WHEN condition THEN
        insert_into_clause [ values_clause ]
 WHEN condition THEN
        insert_into_clause [ values_clause ]
ELSE insert_into_clause [ values_clause ]
```

Tell Me / Show Me

Default Values

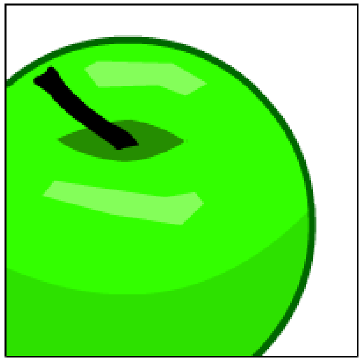
```
CREATE TABLE table1 (
column1          DATE DEFAULT SYSDATE,…)

INSERT INTO table1
   (column1,….)
VALUES
   (DEFAULT,…);
```



Tell Me / Show Me

The Merge Statement

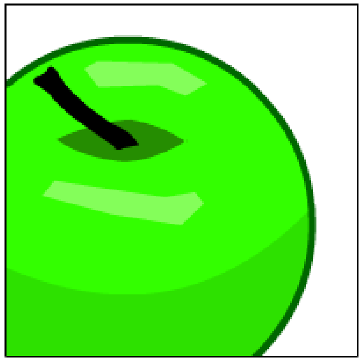
```
MERGE INTO destination-table USING source-table
ON matching-condition
WHEN MATCHED THEN UPDATE
SET ……
WHEN NOT MATCHED THEN INSERT
VALUES (……);
```



ORACLE Academy

Tell Me / Show Me

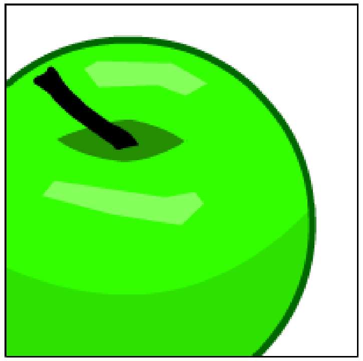
Creating Tables



```
CREATE TABLE table
(column datatype [DEFAULT expression],
column datatype [DEFAULT expression],
……[ ] );

CREATE TABLE tablename
[(column, column, …)]
AS subquery;
```

Tell Me / Show Me

Specifying Data Types



NUMBER(p,s)

CHAR

VARCHAR2(n)

DATE

TIMESTAMP

TIMESTAMP WITH TIMEZONE

TIMESTAMP WITH LOCAL TIME ZONE

INTERVAL YEAR TO MONTH

INTERVAL DAY TO SECOND

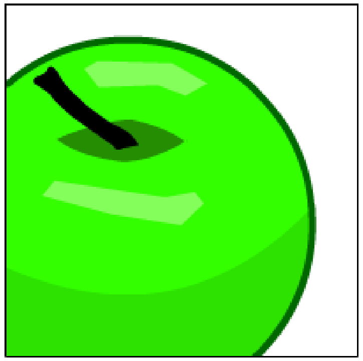
CLOB

BLOB

RAW

Tell Me / Show Me

Modifying a table



```
ALTER TABLE tablename
ADD (column_name datatype [DEFAULT
expression]…)
```

```
ALTER TABLE tablename MODIFY (column_name VARCHAR2(30));
```

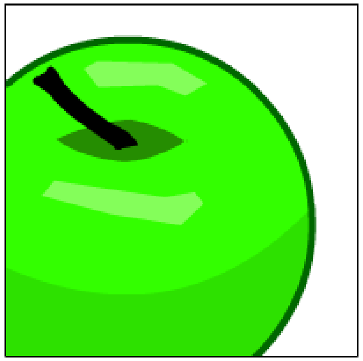
```
ALTER TABLE tablename DROP COLUMN column name;
```

```
ALTER TABLE tablename SET UNUSED (column name);
```

```
ALTER TABLE tablename DROP UNUSED COLUMNS;
```

Tell Me / Show Me

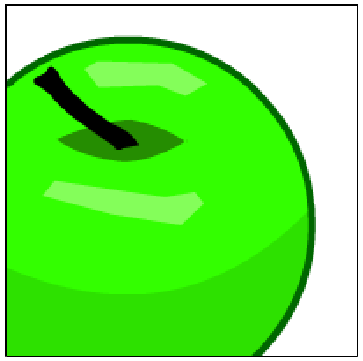
Modifying a table



```
DROP TABLE tablename;

FLASHBACK TABLE tablename TO BEFORE DROP;

SELECT * FROM user_recyclebin;
SELECT versions_starttime "START_DATE",
             versions_endtime   "END_DATE",
             column, column......
FROM   table
  VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE
WHERE  column = value
```

Tell Me / Show Me

Column Level Constraints

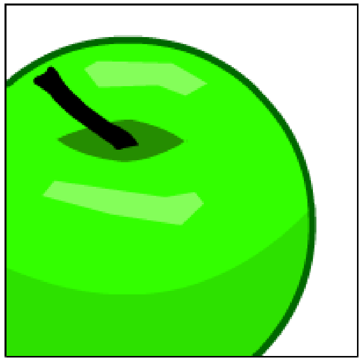


```
CREATE TABLE table
(col1 datatype CONSTRAINT tab_col1_pk PRIMARY KEY,
col2 datatype CONSTRAINT tab_col2_nn NOT NULL,
col3 datatype CONSTRAINT tab_col3_uk UNIQUE,
col4 datatype CONSTRAINT tab_col4_ck CHECK (col4 > value),
col5 datatype CONSTRAINT tab_col5 REFERENCES table2 (col1));
```

Tell Me / Show Me

Table Level Constraints

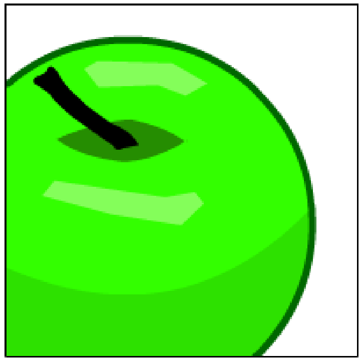
```
CREATE TABLE table
(col1 datatype,
col2 datatype,
col3 datatype,
col4 datatype,
col5 datatype,
CONSTRAINT tab_col1_pk PRIMARY(col1),
CONSTRAINT tab_col3_uk UNIQUE(col2),
 CONSTRAINT tab_col4_ck CHECK (col4 > value),
CONSTRAINT tab1_col5_fk FOREIGN KEY (col5)
REFERENCES table2 (col1));
```



Tell Me / Show Me

Creating and Managing Views

```
CREATE [OR REPLACE] [FORCE| NOFORCE]
VIEW view  [(alias [, alias]...)] AS subquery
[WITH CHECK OPTION [CONSTRAINT constraint]]
[WITH READ ONLY [CONSTRAINT constraint]];

DROP VIEW viewname;
```



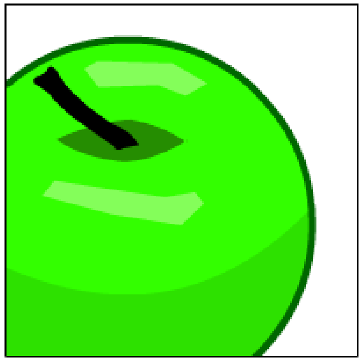
Top-n analysis

```
SELECT ROWNUM as RANK, col1, col2
FROM (SELECT col1, col2 FROM table1
ORDER BY col1)
WHERE ROWNUM <= n;
```

Tell Me / Show Me

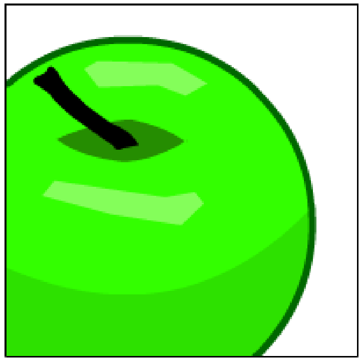
Inline Views

```
SELECT t1.col1, t2.col2…
FROM table 1 t1, (SELECT col1, col2..
                  FROM table2
                  WHERE …) t2
WHERE ……;
```



Tell Me / Show Me

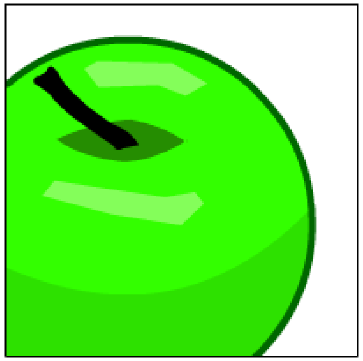
Creating Sequences



```
CREATE SEQUENCE sequence
        [INCREMENT BY n]
        [START WITH n]
        [{MAXVALUE n | NOMAXVALUE}]
        [{MINVALUE n | NOMINVALUE}]
        [{CYCLE | NOCYCLE}]
        [{CACHE n | NOCACHE}];

DROP SEQUENCE sequence_name;
```

Tell Me / Show Me

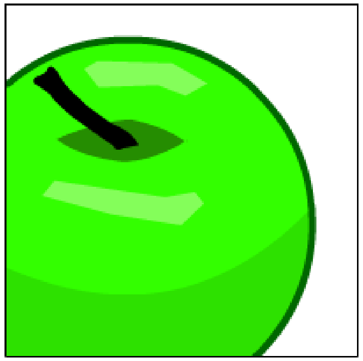
Creating Indexes, and Synonyms



```
CREATE INDEX index_name
ON  table_name( column...,column);

DROP INDEX index_name;

CREATE [PUBLIC] SYNONYM synonym
FOR object;

DROP [PUBLIC] SYNONYM name_of_synonym
```

Tell Me / Show Me

Creating and Revoking Object Privileges

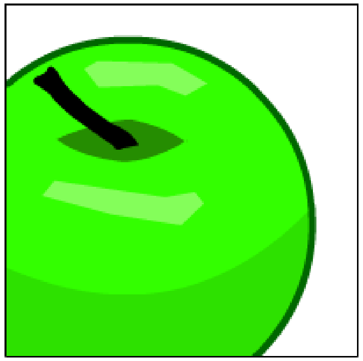


```
CREATE USER user
IDENTIFIED BY   password;

ALTER USER user
IDENTIFIED BY password;

GRANT privilege [, privilege...]
TO user [, user| role, PUBLIC...];
```

ORACLE Academy

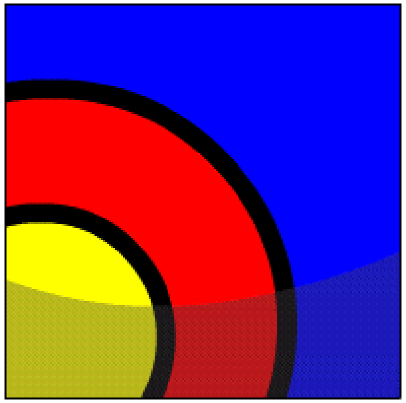
Tell Me / Show Me

Creating and Revoking Object Privileges



```
CREATE ROLE role_name;

GRANT object_priv [(column_list)]
ON object_name
TO {user|role|PUBLIC}
[WITH GRANT OPTION];

REVOKE {privilege [, privilege...]|ALL}
ON  object
FROM   {user[, user...]|role|PUBLIC}
[CASCADE CONSTRAINTS];
```

Summary



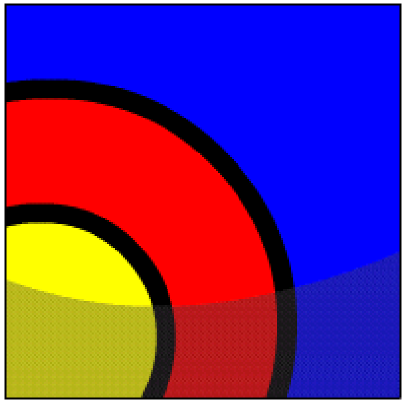
In this lesson you have reviewed:

- The key points about Case and Character Manipulation
- Number, Date, Conversion and General Functions
- Conditional expressions
- Cartesian Product and Join Operations
- Nonequijoins, outer joins, self joins, cross joins, natural joins and join clauses
- Group functions, group by syntax and having clauses
- Single-row and multiple row subqueries

Summary

In this lesson you have reviewed:

- Inserting, updating, and deleting data
- Default values and the merge statement
- Creating tables, specifying data types and modifying a table
- Not null and unique constraints
- Primary key, foreign key and check constraints
- Creating and managing views
- Creating sequences, indexes and synonyms
- Creating and revoking object privileges



ORACLE Academy

Summary

Practice Guide

The link for the lesson practice guide can be found in the course resources in Section 0.