Object	Description
Table	Basic unit of storage; composed of rows and columns
View	Logically represents subsets of data from one or more tables
Sequence	Numeric value generator
Index	Improves the performance of some queries
Synonym	Gives alternative names to objects

Naming Rules

Table names and column names:

- Must begin with a letter
- Must be 1–30 characters long
- Must contain only A–Z, a–z, 0–9, _, $, and #
- Must not duplicate the name of another object owned by the same user
- Must not be an Oracle server reserved word

The CREATE TABLE Statement

You must have:

- CREATE TABLE privilege
- A storage area

```
CREATE TABLE [schema.]table
            (column datatype [DEFAULT expr][, ...])
```

You specify:

- Table name
- Column name, column data type, and column size

Referencing Another User's Tables

Tables belonging to other users are not in the user's schema.

You should use the owner's name as a prefix to those tables.

The DEFAULT Option

Specify a default value for a column during an insert.

```
... hire_date DATE DEFAULT SYSDATE ...
```

Literal values, expressions, or SQL functions are legal values.

Another column's name or a pseudocolumn are illegal values.

The default data type must match the column data type.

ORACLE

Creating Tables

Create the table.

```
CREATE TABLE dept
        (deptno  NUMBER(2),
         dname   VARCHAR2(14),
         loc     VARCHAR2(13));

Table created.
```

Confirm table creation.

```
DESCRIBE dept
```

Name	Null?	Type
DEPTNO		NUMBER(2)
DNAME		VARCHAR2(14)
LOC		VARCHAR2(13)

Querying the Data Dictionary

See the names of tables owned by the user.

```
SELECT table_name
FROM   user_tables
```

View distinct object types owned by the user.

```
SELECT DISTINCT object_type
FROM   user_objects
```

View tables, views, synonyms, and sequences owned by the user.

```
SELECT *
FROM   user_catalog
```

Data Types

Data Type	Description
VARCHAR2 (*size*)	Variable-length character data
CHAR (*size*)	Fixed-length character data
NUMBER (*p*, *s*)	Variable-length numeric data
DATE	Date and time values
LONG	Variable-length character data up to 2 gigabytes
CLOB	Character data up to 4 gigabytes
RAW and LONG RAW	Raw binary data
BLOB	Binary data up to 4 gigabytes
BFILE	Binary data stored in an external file; up to 4 gigabytes
ROWID	A 64 base number system representing the unique address of a row in its table.

DateTime Data Types

Datetime enhancements with Oracle9*i*:

- New Datetime data types have been introduced.
- New data type storage is available.
- Enhancements have been made to time zones and local time zone.

Data Type	Description
TIMESTAMP	Date with fractional seconds
INTERVAL YEAR TO MONTH	Stored as an interval of years and months
INTERVAL DAY TO SECOND	Stored as an interval of days to hours minutes and seconds

ORACLE

Creating a Table by Using a Subquery Syntax

Create a table and insert rows by combining the `CREATE TABLE` statement and the `AS subquery` option.

```
CREATE TABLE table
        [(column, column...)]
AS subquery
```

Match the number of specified columns to the number of subquery columns.

Define columns with column names and default values.

ORACLE

```
CREATE TABLE dept80
AS
  SELECT   employee_id, last_name,
           salary*12 ANNSAL,
           hire_date
  FROM     employees
  WHERE    department id = 80;
able created.
```

```
DESCRIBE dept80
```

Name	Null?	Type
PLOYEE_ID		NUMBER(6)
T_NAME	NOT NULL	VARCHAR2(25)
SAL		NUMBER
E_DATE	NOT NULL	DATE

Adding a New Row to a Table

DEPARTMENTS

New row

...insert a new row into the DEPARTMENTS table...

The INSERT Statement Syntax

Add new rows to a table by using the INSERT statement.

```
INSERT INTO    table [(column [, column...])]
VALUES         (value [, value...]);
```

Only one row is inserted at a time with this syntax.

Inserting New Rows

Insert a new row containing values for each column.

List values in the default order of the columns in the table.

Optionally, list the columns in the INSERT clause.

```
INSERT INTO departments(department_id, department_name,
                        manager_id, location_id)
VALUES          (70, 'Public Relations', 100, 1700);
1 row created
```

Enclose character and date values within single quotation marks.

ORACLE

Inserting Rows with Null Values

Implicit method: Omit the column from the column list.

```
INSERT INTO   departments (department_id,
                           department_name )
VALUES                (30, 'Purchasing' );
1 row created.
```

Explicit method: Specify the NULL keyword in the VALUES clause.

```
INSERT INTO   departments
VALUES                (100, 'Finance', NULL, NULL);
1 row created.
```

Inserting Special Values

The SYSDATE function records the current date and time.

```
INSERT INTO employees (employee_id,
                       first_name, last_name,
                       email, phone_number,
                       hire_date, job_id, salary,
                       commission_pct, manager_id,
                       department_id)
VALUES                 (113,
                        'Louis', 'Popp',
                        'LPOPP', '515.124.4567',
                        SYSDATE, 'AC_ACCOUNT', 6900,
                        NULL, 205, 100);

1 row created.
```

Inserting Specific Date Values

Add a new employee.

```
INSERT INTO employees
VALUES      (114,
             'Den', 'Raphealy',
             'DRAPHEAL', 515.127.4561',
             TO_DATE('FEB 3, 1999', 'MON DD, YYYY'),
             'AC_ACCOUNT', 11000, NULL, 100, 30);

1 row created.
```

Verify your addition.

Creating a Script

Use & substitution in a SQL statement to prompt for values.

& is a placeholder for the variable value.

```
INSERT INTO departments
            (department_id, department_name, location_id)
VALUES      (&department_id, '&department_name', &location);
```

Define Substitution Variables

"department_id" 40

"department_name" Human Resources

"location" 3500

Submit for Execution | Cancel

```
1 row created.
```

Copying Rows from Another Table

Write your INSERT statement with a subquery.

```
INSERT INTO sales_reps(id, name, salary, commission_pct)
  SELECT employee_id, last_name, salary, commission_pct
  FROM   employees
  WHERE  job_id LIKE '%REP%';

4 rows created
```

Do not use the VALUES clause.

Match the number of columns in the INSERT clause to those in the subquery.