Appendix C

SQL Statement Syntax

SQL Statements

This excerpt from the Oracle Database SQL Quick Reference guide presents the syntax for Oracle SQL statements. SQL statements are the means by which programs and users access data in an Oracle database.

Table 1 shows each SQL statement and its related syntax. Table 2 shows the syntax of the subclauses found in the table 1.

See Also: Oracle Database SQL Reference for detailed information about Oracle SQL

Table 1: Syntax for SQL Statements

SQL Statement	Syntax
ALTER CLUSTER	ALTER CLUSTER [schema.]cluster { physical_attributes_clause | SIZE size_clause | allocate_extent_clause | deallocate_unused_clause | { CACHE | NOCACHE } } [physical_attributes_clause | SIZE size_clause | allocate_extent_clause | deallocate_unused_clause | { CACHE | NOCACHE }]... [parallel_clause] ;
ALTER DATABASE	ALTER DATABASE [database] { startup_clauses | recovery_clauses | database_file_clauses | logfile_clauses | controlfile_clauses | standby_database_clauses | default_settings_clauses | redo_thread_clauses | security_clause } ;
ALTER DIMENSION	ALTER DIMENSION [schema.]dimension { ADD { level_clause | hierarchy_clause | attribute_clause | extended_attribute_clause } [ADD

SQL Statement	Syntax
	{ level_clause | hierarchy_clause | attribute_clause | extended_attribute_clause }]... | DROP { LEVEL level [RESTRICT | CASCADE] | HIERARCHY hierarchy | ATTRIBUTE attribute [LEVEL level [COLUMN column [, COLUMN column]...] } [DROP { LEVEL level [RESTRICT | CASCADE] | HIERARCHY hierarchy | ATTRIBUTE attribute [LEVEL level [COLUMN column [, COLUMN column]...] }]... | COMPILE } ;
ALTER DISKGROUP	ALTER DISKGROUP { disk_clauses | diskgroup_clauses } [{ disk_clauses | diskgroup_clauses }]... ;
ALTER FUNCTION	ALTER FUNCTION [schema.]function COMPILE [DEBUG] [compiler_parameters_clause [compiler_parameters_clause] ...] [REUSE SETTINGS] ;
ALTER INDEX	ALTER INDEX [schema.]index { { deallocate_unused_clause | allocate_extent_clause | shrink_clause | parallel_clause | physical_attributes_clause | logging_clause } [deallocate_unused_clause | allocate_extent_clause | shrink_clause | parallel_clause | physical_attributes_clause | logging_clause]... | rebuild_clause | PARAMETERS ('ODCI_parameters')

SQL Statement	Syntax
		{ ENABLE	DISABLE } | UNUSABLE | RENAME TO new_name | COALESCE | { MONITORING	NOMONITORING } USAGE | UPDATE BLOCK REFERENCES | alter_index_partitioning } ;
ALTER INDEXTYPE	ALTER INDEXTYPE [schema.]indextype { { ADD	DROP } [schema.]operator (parameter_types) [, { ADD	DROP } [schema.]operator (parameter_types)]... [using_type_clause] 	COMPILE } ;
ALTER JAVA	ALTER JAVA { SOURCE	CLASS } [schema.]object_name [RESOLVER ((match_string [,] { schema_name | - }) [(match_string [,] { schema_name | - })]...)] { { COMPILE	RESOLVE } 	invoker_rights_clause } ;
ALTER MATERIALIZED VIEW	ALTER MATERIALIZED VIEW [schema.](materialized_view) [physical_attributes_clause | table_compression | LOB_storage_clause [, LOB_storage_clause]... | modify_LOB_storage_clause [, modify_LOB_storage_clause]... | alter_table_partitioning | parallel_clause | logging_clause | allocate_extent_clause | shrink_clause | { CACHE	NOCACHE }] [alter_iot_clauses] [USING INDEX physical_attributes_clause] [MODIFY scoped_table_ref_constraint | alter_mv_refresh] [{ ENABLE	DISABLE } QUERY REWRITE | COMPILE

SQL Statement	Syntax
		CONSIDER FRESH] ;
ALTER MATERIALIZED VIEW LOG	ALTER MATERIALIZED VIEW LOG [FORCE] ON [schema.]table [physical_attributes_clause | alter_table_partitioning | parallel_clause | logging_clause | allocate_extent_clause | shrink_clause | { CACHE | NOCACHE }] [ADD { { OBJECT ID | PRIMARY KEY | ROWID | SEQUENCE } [(column [, column]...)] | (column [, column]...) } [, { { OBJECT ID | PRIMARY KEY | ROWID | SEQUENCE } [(column [, column]...)] | (column [, column]...) }]... [new_values_clause]] ;
ALTER OPERATOR	ALTER OPERATOR [schema.]operator { add_binding_clause | drop_binding_clause | COMPILE } ;
ALTER OUTLINE	ALTER OUTLINE [PUBLIC | PRIVATE] outline { REBUILD | RENAME TO new_outline_name | CHANGE CATEGORY TO new_category_name | { ENABLE | DISABLE } } [REBUILD | RENAME TO new_outline_name | CHANGE CATEGORY TO new_category_name | { ENABLE | DISABLE }]... ;

SQL Statement	Syntax
ALTER PACKAGE	ALTER PACKAGE [schema.]package COMPILE [DEBUG] [PACKAGE | SPECIFICATION | BODY] [compiler_parameters_clause [compiler_parameters_clause] ...] [REUSE SETTINGS] ;
ALTER PROCEDURE	ALTER PROCEDURE [schema.]procedure COMPILE [DEBUG] [compiler_parameters_clause [compiler_parameters_clause] ...] [REUSE SETTINGS] ;
ALTER PROFILE	ALTER PROFILE profile LIMIT { resource_parameters | password_parameters } [resource_parameters | password_parameters]... ;
ALTER RESOURCE COST	ALTER RESOURCE COST { CPU_PER_SESSION | CONNECT_TIME | LOGICAL_READS_PER_SESSION | PRIVATE_SGA } integer [{ CPU_PER_SESSION | CONNECT_TIME | LOGICAL_READS_PER_SESSION | PRIVATE_SGA } integer] ... ;
ALTER ROLE	ALTER ROLE role { NOT IDENTIFIED | IDENTIFIED { BY password | USING [schema.]package | EXTERNALLY | GLOBALLY } } ;
ALTER ROLEBACK SEGMENT	ALTER ROLLBACK SEGMENT rollback_segment { ONLINE | OFFLINE | storage_clause | SHRINK [TO integer [K | M]] };

SQL Statement	Syntax
ALTER SEQUENCE	ALTER SEQUENCE [schema.]sequence { INCREMENT BY integer | { MAXVALUE integer | NOMAXVALUE } | { MINVALUE integer | NOMINVALUE } | { CYCLE | NOCYCLE } | { CACHE integer | NOCACHE } | { ORDER | NOORDER } } [INCREMENT BY integer | { MAXVALUE integer | NOMAXVALUE } | { MINVALUE integer | NOMINVALUE } | { CYCLE | NOCYCLE } | { CACHE integer | NOCACHE } | { ORDER | NOORDER }]... ;
ALTER SESSION	ALTER SESSION { ADVISE { COMMIT | ROLLBACK | NOTHING } | CLOSE DATABASE LINK dblink | { ENABLE | DISABLE } COMMIT IN PROCEDURE | { ENABLE | DISABLE } GUARD | { ENABLE | DISABLE | FORCE } PARALLEL { DML | DDL | QUERY } [PARALLEL integer] | { ENABLE RESUMABLE [TIMEOUT integer] [NAME string] | DISABLE RESUMABLE } | alter_session_set_clause } ;
ALTER SYSTEM	ALTER SYSTEM { archive_log_clause | checkpoint_clause | check_datafiles_clause | DUMP ACTIVE SESSION HISTORY [MINUTES integer] | distributed_recov_clauses | restricted_session_clauses | FLUSH { SHARED_POOL | BUFFER_CACHE } | end_session_clauses | SWITCH LOGFILE | { SUSPEND | RESUME } | quiesce_clauses | shutdown_dispatcher_clause | REGISTER | SET alter_system_set_clause [alter_system_set_clause]... | RESET alter_system_reset_clause [alter_system_reset_clause]... } ;
ALTER TABLE	ALTER TABLE [schema.]table [alter_table_properties | column_clauses

SQL Statement	Syntax
		constraint_clauses | alter_table_partitioning | alter_external_table_clauses | move_table_clause] [enable_disable_clause | { ENABLE	DISABLE } { TABLE LOCK	ALL TRIGGERS } [enable_disable_clause | { ENABLE	DISABLE } { TABLE LOCK	ALL TRIGGERS }]...] ;
ALTER TABLESPACE	ALTER TABLESPACE tablespace { DEFAULT [table_compression] storage_clause | MINIMUM EXTENT integer [K	M] | RESIZE size_clause | COALESCE | RENAME TO new_tablespace_name | { BEGIN	END } BACKUP | datafile_tempfile_clauses | tablespace_logging_clauses | tablespace_group_clause | tablespace_state_clauses | autoextend_clause | flashback_mode_clause | tablespace_retention_clause } ;
ALTER TRIGGER	ALTER TRIGGER [schema.]trigger { ENABLE | DISABLE | RENAME TO new_name | COMPILE [DEBUG] [compiler_parameters_clause [compiler_parameters_clause] ...] [REUSE SETTINGS] } ;
ALTER TYPE	ALTER TYPE [schema.]type { compile_type_clause 	replace_type_clause 	{ alter_method_spec 	alter_attribute_definition 	alter_collection_clauses 	[NOT] { INSTANTIABLE	FINAL } } [dependent_handling_clause] } ;

SQL Statement	Syntax
ALTER USER	ALTER USER { user { IDENTIFIED { BY password [REPLACE old_password] | EXTERNALLY | GLOBALLY AS 'external_name' } | DEFAULT TABLESPACE tablespace | TEMPORARY TABLESPACE { tablespace | tablespace_group_name } | QUOTA { integer [K | M] | UNLIMITED } ON tablespace [QUOTA { integer [K | M] | UNLIMITED } ON tablespace]... | PROFILE profile | DEFAULT ROLE { role [, role]... | ALL [EXCEPT role [, role]...] | NONE } | PASSWORD EXPIRE | ACCOUNT { LOCK | UNLOCK } } [{ IDENTIFIED { BY password [REPLACE old_password] | EXTERNALLY | GLOBALLY AS 'external_name' } | DEFAULT TABLESPACE tablespace | TEMPORARY TABLESPACE { tablespace | tablespace_group_name } | QUOTA { integer [K | M] | UNLIMITED } ON tablespace [QUOTA { integer [K | M] | UNLIMITED } ON tablespace]... | PROFILE profile | DEFAULT ROLE { role [, role]... | ALL [EXCEPT role [, role]...] | NONE } | PASSWORD EXPIRE | ACCOUNT { LOCK | UNLOCK } }]... | user [, user]... proxy_clause ;

SQL Statement	Syntax
ALTER VIEW	ALTER VIEW [schema.]view { ADD out_of_line_constraint | MODIFY CONSTRAINT constraint { RELY | NORELY } | DROP { CONSTRAINT constraint | PRIMARY KEY | UNIQUE (column [, column]...) } | COMPILE } ;
ANALYZE	ANALYZE { TABLE [schema.]table [PARTITION (partition) | SUBPARTITION (subpartition)] | INDEX [schema.]index [PARTITION (partition) | SUBPARTITION (subpartition)] | CLUSTER [schema.]cluster } { validation_clauses | LIST CHAINED ROWS [into_clause] | DELETE [SYSTEM] STATISTICS | compute_statistics_clause | estimate_statistics_clause } ;
ASSOCIATE STATISTICS	ASSOCIATE STATISTICS WITH { column_association | function_association } ;
AUDIT	AUDIT { sql_statement_clause | schema_object_clause } [BY { SESSION | ACCESS }] [WHENEVER [NOT] SUCCESSFUL] ;
CALL	CALL { routine_clause | object_access_expression } [INTO :host_variable [[INDICATOR] :indicator_variable]] ;
COMMENT	COMMENT ON { TABLE [schema.] { table | view } | COLUMN [schema.] { table. | view. | materialized_view. } column | OPERATOR [schema.] operator | INDEXTYPE [schema.] indextype

SQL Statement	Syntax
		MATERIALIZED VIEW materialized_view } IS 'text' ;
COMMIT	COMMIT [WORK] [COMMENT 'text' | FORCE 'text' [, integer]] ;
CREATE CLUSTER	CREATE CLUSTER [schema.]cluster (column datatype [SORT] [, column datatype [SORT]]...) [{ physical_attributes_clause | SIZE size_clause | TABLESPACE tablespace | { INDEX | [SINGLE TABLE] HASHKEYS integer [HASH IS expr] } } [physical_attributes_clause | SIZE size_clause | TABLESPACE tablespace | { INDEX | [SINGLE TABLE] HASHKEYS integer [HASH IS expr] }]...] [parallel_clause] [NOROWDEPENDENCIES	ROWDEPENDENCIES] [CACHE	NOCACHE] ;
CREATE CONTEXT	CREATE [OR REPLACE] CONTEXT namespace USING [schema.] package [INITIALIZED { EXTERNALLY	GLOBALLY } 	ACCESSED GLOBALLY] ;
CREATE CONTROLFILE	CREATE CONTROLFILE [REUSE] [SET] DATABASE database [logfile_clause] { RESETLOGS	NORESETLOGS } [DATAFILE file_specification [, file_specification]...] [{ MAXLOGFILES integer | MAXLOGMEMBERS integer | MAXLOGHISTORY integer

SQL Statement	Syntax
		MAXDATAFILES integer | MAXINSTANCES integer | { ARCHIVELOG	NOARCHIVELOG } | FORCE LOGGING } [MAXLOGFILES integer | MAXLOGMEMBERS integer | MAXLOGHISTORY integer | MAXDATAFILES integer | MAXINSTANCES integer | { ARCHIVELOG	NOARCHIVELOG } | FORCE LOGGING]...] [character_set_clause] ;
CREATE DATABASE	CREATE DATABASE [database] { USER SYS IDENTIFIED BY password | USER SYSTEM IDENTIFIED BY password | CONTROLFILE REUSE | MAXDATAFILES integer | MAXINSTANCES integer | CHARACTER SET charset | NATIONAL CHARACTER SET charset | SET DEFAULT { BIGFILE	SMALLFILE } TABLESPACE | database_logging_clauses | tablespace_clauses | set_time_zone_clause }... ;
CREATE DATABASE LINK	CREATE [SHARED] [PUBLIC] DATABASE LINK dblink [CONNECT TO { CURRENT_USER | user IDENTIFIED BY password [dblink_authentication] } | dblink_authentication] [USING 'connect_string'] ;
CREATE DIMENSION	CREATE DIMENSION [schema.]dimension level_clause [level_clause]... { hierarchy_clause | attribute_clause | extended_attribute_clause } [hierarchy_clause | attribute_clause | extended_attribute_clause]... ;

SQL Statement	Syntax
CREATE DIRECTORY	CREATE [OR REPLACE] DIRECTORY directory AS 'path_name' ;

CREATE DISKGROUP	CREATE DISKGROUP diskgroup_name
	  [{ HIGH | NORMAL | EXTERNAL } REDUNDANCY]
	  [FAILGROUP failgroup_name]
	  DISK qualified_disk_clause
	  [, qualified_disk_clause]...
	  [[FAILGROUP failgroup_name]
	  DISK qualified_disk_clause
	  [, qualified_disk_clause]...
	  ]... ;

CREATE FUNCTION	CREATE [OR REPLACE] FUNCTION [schema.]function
	  [(argument [IN | OUT | IN OUT]
	  [NOCOPY] datatype
	  [, argument [IN | OUT | IN OUT]
	  [NOCOPY] datatype
	  ]...
	  )
	  ]
	  RETURN datatype
	  [{ invoker_rights_clause
	  | DETERMINISTIC
	  | parallel_enable_clause
	  }
	  [invoker_rights_clause
	  | DETERMINISTIC
	  | parallel_enable_clause
	  ]...
	  ]
	  { { AGGREGATE | PIPELINED }
	  USING [schema.]implementation_type
	  | [PIPELINED]
	  { IS | AS }
	  { pl/sql_function_body | call_spec }
	  } ;

CREATE INDEX	CREATE [UNIQUE | BITMAP] INDEX [schema.]index
	  ON { cluster_index_clause
	  | table_index_clause
	  | bitmap_join_index_clause
	  } ;

CREATE INDEXTYPE	CREATE [OR REPLACE] INDEXTYPE
	  [schema.]indextype FOR
	  [schema.]operator (paramater_type
	  [, paramater_type]...)
	  [, [schema.]operator (paramater_type
	  [, paramater_type]...)
	  ]...
	  using_type_clause ;

SQL Statement	Syntax
CREATE JAVA	CREATE [OR REPLACE] [AND { RESOLVE | COMPILE }] [NOFORCE] JAVA { { SOURCE | RESOURCE } NAMED [schema.]primary_name | CLASS [SCHEMA schema] } [invoker_rights_clause] [RESOLVER ((match_string [,] { schema_name | - }) [(match_string [,] { schema_name | - })]...)] { USING { BFILE (directory_object_name , server_file_name) | { CLOB | BLOB | BFILE } subquery | 'key_for_BLOB' } | AS source_text } ;
CREATE LIBRARY	CREATE [OR REPLACE] LIBRARY [schema.]libname { IS | AS } 'filename' [AGENT 'agent_dblink'] ;
CREATE MATERIALIZED VIEW	CREATE MATERIALIZED VIEW [schema.]materialized_view [OF [schema.]object_type] [(scoped_table_ref_constraint)] { ON PREBUILT TABLE [{ WITH | WITHOUT } REDUCED PRECISION] | physical_properties materialized_view_props } [USING INDEX [physical_attributes_clause | TABLESPACE tablespace] [physical_attributes_clause | TABLESPACE tablespace]... | USING NO INDEX] [create_mv_refresh] [FOR UPDATE] [{ DISABLE | ENABLE } QUERY REWRITE] AS subquery ;
CREATE MATERIALIZED VIEW LOG	CREATE MATERIALIZED VIEW LOG ON [schema.] table [physical_attributes_clause | TABLESPACE tablespace

SQL Statement	Syntax
		logging_clause | { CACHE	NOCACHE } [physical_attributes_clause | TABLESPACE tablespace | logging_clause | { CACHE	NOCACHE }]...] [parallel_clause] [table_partitioning_clauses] [WITH { OBJECT ID | PRIMARY KEY | ROWID | SEQUENCE | (column [, column]...) } [, { OBJECT ID | PRIMARY KEY | ROWID | SEQUENCE | (column [, column]...) }]... [new_values_clause]] ;
CREATE OPERATOR	CREATE [OR REPLACE] OPERATOR [schema.] operator binding_clause ;
CREATE OUTLINE	CREATE [OR REPLACE] [PUBLIC | PRIVATE] OUTLINE [outline] [FROM [PUBLIC | PRIVATE] source_outline] [FOR CATEGORY category] [ON statement] ;
CREATE PACKAGE	CREATE [OR REPLACE] PACKAGE [schema.]package [invoker_rights_clause] { IS | AS } pl/sql_package_spec ;
CREATE PACKAGE BODY	CREATE [OR REPLACE] PACKAGE BODY [schema.]package { IS | AS } pl/sql_package_body ;
CREATE PFILE	CREATE PFILE [= 'pfile_name'] FROM SPFILE [= 'spfile_name'] ;
CREATE PROCEDURE	CREATE [OR REPLACE] PROCEDURE [schema.]procedure [(argument [IN | OUT | IN OUT] [NOCOPY] datatype [, argument [IN | OUT | IN OUT] [NOCOPY]

SQL Statement	Syntax
	datatype]...)] [invoker_rights_clause] { IS | AS } { pl/sql_subprogram_body | call_spec } ;
CREATE PROFILE	CREATE PROFILE profile LIMIT { resource_parameters | password_parameters } [resource_parameters | password_parameters]... ;
CREATE ROLE	CREATE ROLE role [NOT IDENTIFIED | IDENTIFIED { BY password | USING [schema.] package | EXTERNALLY | GLOBALLY }] ;
CREATE ROLEBACK SEGMENT	CREATE [PUBLIC] ROLLBACK SEGMENT rollback_segment [{ TABLESPACE tablespace | storage_clause } [TABLESPACE tablespace | storage_clause]...];
CREATE SCHEMA	CREATE SCHEMA AUTHORIZATION schema { create_table_statement | create_view_statement | grant_statement } [create_table_statement | create_view_statement | grant_statement]... ;
CREATE SEQUENCE	CREATE SEQUENCE [schema.]sequence [{ INCREMENT BY | START WITH } integer | { MAXVALUE integer | NOMAXVALUE } | { MINVALUE integer | NOMINVALUE } | { CYCLE | NOCYCLE } | { CACHE integer | NOCACHE } | { ORDER | NOORDER }] [{ INCREMENT BY | START WITH } integer | { MAXVALUE integer | NOMAXVALUE } | { MINVALUE integer | NOMINVALUE } | { CYCLE | NOCYCLE }

SQL Statement	Syntax
		{ CACHE integer | NOCACHE } | { ORDER | NOORDER }]... ;
CREATE SPFILE	CREATE SPFILE [= 'spfile_name'] FROM PFILE [= 'pfile_name'] ;
CREATE SYNONYM	CREATE [OR REPLACE] [PUBLIC] SYNONYM [schema.]synonym FOR [schema.]object [@ dblink] ;
CREATE TABLE	{ relational_table | object_table | XMLType_table }
CREATE TABLESPACE	CREATE [BIGFILE | SMALLFILE] { permanent_tablespace_clause | temporary_tablespace_clause | undo_tablespace_clause } ;
CREATE TRIGGER	CREATE [OR REPLACE] TRIGGER [schema.]trigger { BEFORE | AFTER | INSTEAD OF } { dml_event_clause | { ddl_event [OR ddl_event]... | database_event [OR database_event]... } ON { [schema.]SCHEMA | DATABASE } } [WHEN (condition)] { pl/sql_block | call_procedure_statement } ;
CREATE TYPE	{ create_incomplete_type | create_object_type | create_varray_type | create_nested_table_type }

CREATE TYPE BODY	CREATE [OR REPLACE] TYPE BODY [schema.]type_name
	{ IS | AS }
	{ subprogram_declaration
	| map_order_func_declaration
	}
	[; { subprogram_declaration
	| map_order_func_declaration
	}
	]...
	END ;

SQL Statement	Syntax
CREATE USER	CREATE USER user IDENTIFIED { BY password | EXTERNALLY | GLOBALLY AS 'external_name' } [DEFAULT TABLESPACE tablespace | TEMPORARY TABLESPACE { tablespace | tablespace_group_name } | QUOTA { integer [K | M] | UNLIMITED } ON tablespace [QUOTA { integer [K | M] | UNLIMITED } ON tablespace]... | PROFILE profile | PASSWORD EXPIRE | ACCOUNT { LOCK | UNLOCK } [DEFAULT TABLESPACE tablespace | TEMPORARY TABLESPACE { tablespace | tablespace_group_name } | QUOTA { integer [K | M] | UNLIMITED } ON tablespace [QUOTA { integer [K | M] | UNLIMITED } ON tablespace]... | PROFILE profile | PASSWORD EXPIRE | ACCOUNT { LOCK | UNLOCK }]...] ;
CREATE VIEW	CREATE [OR REPLACE] [[NO] FORCE] VIEW [schema.]view [(alias [inline_constraint [inline_constraint]...] | out_of_line_constraint [, alias [inline_constraint [inline_constraint]...] | out_of_line_constraint]...) | object_view_clause | XMLType_view_clause] AS subquery [subquery_restriction_clause] ;

SQL Statement	Syntax
DELETE	DELETE [hint] [FROM] { dml_table_expression_clause | ONLY (dml_table_expression_clause) } [t_alias] [where_clause] [returning_clause] ;
DISASSOCIATE STATISTICS	DISASSOCIATE STATISTICS FROM { COLUMNS [schema.]table.column [, [schema.]table.column]... | FUNCTIONS [schema.]function [, [schema.]function]... | PACKAGES [schema.]package [, [schema.]package]... | TYPES [schema.]type [, [schema.]type]... | INDEXES [schema.]index [, [schema.]index]... | INDEXTYPES [schema.]indextype [, [schema.]indextype]... } [FORCE] ;
DROP CLUSTER	DROP CLUSTER [schema.]cluster [INCLUDING TABLES [CASCADE CONSTRAINTS]] ;
DROP CONTEXT	DROP CONTEXT namespace ;
DROP DATABASE	DROP DATABASE ;
DROP DATABASE LINK	DROP [PUBLIC] DATABASE LINK dblink ;
DROP DIMENSION	DROP DIMENSION [schema.]dimension ;
DROP DIRECTORY	DROP DIRECTORY directory_name ;
DROP DISKGROUP	DROP DISKGROUP diskgroup_name [{ INCLUDING | EXCLUDING } CONTENTS] ;
DROP FUNCTION	DROP FUNCTION [schema.]function_name ;
DROP INDEX	DROP INDEX [schema.]index [FORCE] ;

SQL Statement	Syntax
DROP INDEXTYPE	DROP INDEXTYPE [schema.]indextype [FORCE] ;
DROP JAVA	DROP JAVA { SOURCE | CLASS | RESOURCE } [schema.]object_name ;
DROP LIBRARY	DROP LIBRARY library_name ;
DROP MATERIALIZED VIEW	DROP MATERIALIZED VIEW [schema.]materialized_view [PRESERVE TABLE] ;
DROP MATERIALIZED VIEW LOG	DROP MATERIALIZED VIEW LOG ON [schema.]table ;
DROP OPERATOR	DROP OPERATOR [schema.]operator [FORCE] ;
DROP OUTLINE	DROP OUTLINE outline ;
DROP PACKAGE	DROP PACKAGE [BODY] [schema.]package ;
DROP PROCEDURE	DROP PROCEDURE [schema.]procedure ;
DROP PROFILE	DROP PROFILE profile [CASCADE] ;
DROP ROLE	DROP ROLE role ;
DROP ROLLBACK SEGMENT	DROP ROLLBACK SEGMENT rollback_segment ;
DROP SEQUENCE	DROP SEQUENCE [schema.]sequence_name ;
DROP SYNONYM	DROP [PUBLIC] SYNONYM [schema.]synonym [FORCE] ;
DROP TABLE	DROP TABLE [schema.]table [CASCADE CONSTRAINTS] [PURGE] ;
DROP TABLESPACE	DROP TABLESPACE tablespace [INCLUDING CONTENTS [AND DATAFILES] [CASCADE CONSTRAINTS]] ;
DROP TRIGGER	DROP TRIGGER [schema.]trigger ;

SQL Statement	Syntax
DROP TYPE	DROP TYPE [schema.]type_name [FORCE | VALIDATE] ;
DROP TYPE BODY	DROP TYPE BODY [schema.]type_name ;
DROP USER	DROP USER user [CASCADE] ;
DROP VIEW	DROP VIEW [schema.] view [CASCADE CONSTRAINTS] ;
EXPLAIN PLAN	EXPLAIN PLAN [SET STATEMENT_ID = 'text'] [INTO [schema.]table [@ dblink]] FOR statement ;
FLASHBACK DATABASE	FLASHBACK [STANDBY] DATABASE [database] { TO { SCN | TIMESTAMP } expr | TO BEFORE { SCN | TIMESTAMP } expr };
FLASHBACK TABLE	FLASHBACK TABLE [schema.]table [, [schema.]table]... TO { { SCN | TIMESTAMP } expr [{ ENABLE | DISABLE } TRIGGERS] | BEFORE DROP [RENAME TO table] } ;
GRANT	GRANT { grant_system_privileges | grant_object_privileges } ;
INSERT	INSERT [hint] { single_table_insert | multi_table_insert } ;
LOCK TABLE	LOCK TABLE [schema.] { table | view } [{ PARTITION (partition) | SUBPARTITION (subpartition) } | @ dblink] [, [schema.] { table | view } [{ PARTITION (partition) | SUBPARTITION (subpartition) } | @ dblink]

SQL Statement	Syntax
	]... IN lockmode MODE [NOWAIT] ;
MERGE	MERGE [hint] INTO [schema.]table [t_alias] USING [schema.] { table | view | subquery } [t_alias] ON (condition) [merge_update_clause] [merge_insert_clause] ;
NOAUDIT	NOAUDIT { sql_statement_clause [, sql_statement_clause]... | schema_object_clause [, schema_object_clause]... } [WHENEVER [NOT] SUCCESSFUL] ;
PURGE	PURGE { { TABLE table | INDEX index } | { RECYCLEBIN | DBA_RECYCLEBIN } | TABLESPACE tablespace [USER user] } ;
RENAME	RENAME old_name TO new_name ;
REVOKE	REVOKE { revoke_system_privileges | revoke_object_privileges } ;
ROLLBACK	ROLLBACK [WORK] [TO [SAVEPOINT] savepoint | FORCE 'text'] ;
SAVEPOINT	SAVEPOINT savepoint ;
SELECT	subquery [for_update_clause] ;
SET CONSTRAINT[S]	SET { CONSTRAINT | CONSTRAINTS } { constraint [, constraint]... | ALL } { IMMEDIATE | DEFERRED } ;

SQL Statement	Syntax
SET ROLE	SET ROLE { role [IDENTIFIED BY password] [, role [IDENTIFIED BY password]]... | ALL [EXCEPT role [, role]...] | NONE } ;
SET TRANSACTION	SET TRANSACTION { { READ { ONLY | WRITE } | ISOLATION LEVEL { SERIALIZABLE | READ COMMITTED } | USE ROLLBACK SEGMENT rollback_segment } [NAME 'text'] | NAME 'text' } ;
TRUNCATE	TRUNCATE { TABLE [schema.]table [{ PRESERVE | PURGE } MATERIALIZED VIEW LOG] | CLUSTER [schema.]cluster } [{ DROP | REUSE } STORAGE] ;
UPDATE	UPDATE [hint] { dml_table_expression_clause | ONLY (dml_table_expression_clause) } [t_alias] update_set_clause [where_clause] [returning_clause] ;

Table 2: Syntax for Subclauses

Subclause	Syntax
activate_standby_db_clause	ACTIVATE [PHYSICAL | LOGICAL] STANDBY DATABASE [SKIP [STANDBY LOGFILE]]
add_binding_clause	ADD BINDING (parameter_type [, parameter_type]...) RETURN (return_type) [implementation_clause] using_function_clause

Subclause	Syntax
add_column_clause	ADD (column datatype [DEFAULT expr] [{ inline_constraint [inline_constraint]... | inline_ref_constraint }] [, column datatype [DEFAULT expr] [{ inline_constraint [inline_constraint]... | inline_ref_constraint }]]...) [column_properties]
add_disk_clause	ADD [FAILGROUP failgroup_name] DISK qualified_disk_clause [, qualified_disk_clause]... [[FAILGROUP failgroup_name] DISK qualified_disk_clause [, qualified_disk_clause]...]...
add_hash_index_partition	ADD PARTITION [partition_name] [TABLESPACE tablespace_name] [parallel_clause]
add_hash_partition_clause	ADD PARTITION [partition] partitioning_storage_clause [update_index_clauses] [parallel_clause]
add_hash_subpartition	ADD subpartition_spec [update_index_clauses] [parallel_clause]
add_list_partition_clause	ADD PARTITION [partition] list_values_clause [table_partition_description] [update_index_clauses]
add_list_subpartition	ADD subpartition_spec [update_index_clauses]

Subclause	Syntax
add_logfile_clauses	ADD [STANDBY] LOGFILE { [INSTANCE 'instance_name' | THREAD integer] [GROUP integer] redo_log_file_spec [, [GROUP integer] redo_log_file_spec]... | MEMBER 'filename' [REUSE] [, 'filename' [REUSE]]... TO logfile_descriptor [, logfile_descriptor]... }
add_overflow_clause	ADD OVERFLOW [segment_attributes_clause] [(PARTITION [segment_attributes_clause] [, PARTITION [segment_attributes_clause]]...)]
add_range_partition_clause	ADD PARTITION [partition] range_values_clause [table_partition_description] [update_index_clauses]
add_table_partition	{ add_range_partition_clause | add_hash_partition_clause | add_list_partition_clause }
alias_file_name	+diskgroup_name [(template_name)] /alias_name
allocate_extent_clause	ALLOCATE EXTENT [({ SIZE size_clause | DATAFILE 'filename' | INSTANCE integer } [SIZE size_clause | DATAFILE 'filename' | INSTANCE integer]...)]
alter_attribute_definition	{ { ADD | MODIFY } ATTRIBUTE { attribute [datatype] | (attribute datatype [, attribute datatype]...) } | DROP ATTRIBUTE

Subclause	Syntax
	{ attribute | (attribute [, attribute]...) } }
alter_collection_clauses	MODIFY { LIMIT integer | ELEMENT TYPE datatype }
alter_datafile_clause	DATAFILE { 'filename' | filenumber } [, 'filename' | filenumber]... } { ONLINE | OFFLINE [FOR DROP] | RESIZE size_clause | autoextend_clause | END BACKUP }
alter_external_table_clauses	{ add_column_clause | modify_column_clauses | drop_column_clause | parallel_clause | external_data_properties | REJECT LIMIT { integer | UNLIMITED } | PROJECT COLUMN { ALL | REFERENCED } } [add_column_clause | modify_column_clauses | drop_column_clause | parallel_clause | external_data_properties | REJECT LIMIT { integer | UNLIMITED } | PROJECT COLUMN { ALL | REFERENCED }]...
alter_index_partitioning	{ modify_index_default_attrs | add_hash_index_partition | modify_index_partition | rename_index_partition | drop_index_partition | split_index_partition | coalesce_index_partition | modify_index_subpartition }
alter_iot_clauses	{ index_org_table_clause | alter_overflow_clause | alter_mapping_table_clauses | COALESCE }

Subclause	Syntax
alter_mapping_table_clauses	MAPPING TABLE { UPDATE BLOCK REFERENCES | allocate_extent_clause | deallocate_unused_clause }
alter_method_spec	{ ADD | DROP } { map_order_function_spec | subprogram_spec } [{ ADD | DROP } { map_order_function_spec | subprogram_spec }]...
alter_mv_refresh	REFRESH { { FAST | COMPLETE | FORCE } | ON { DEMAND | COMMIT } | { START WITH | NEXT } date | WITH PRIMARY KEY | USING { DEFAULT MASTER ROLLBACK SEGMENT | MASTER ROLLBACK SEGMENT rollback_segment } | USING { ENFORCED | TRUSTED } CONSTRAINTS }
alter_overflow_clause	{ OVERFLOW { allocate_extent_clause | deallocate_unused_clause } [allocate_extent_clause | deallocate_unused_clause]... | add_overflow_clause }
alter_session_set_clause	SET parameter_name = parameter_value [parameter_name = parameter_value]...
alter_system_reset_clause	parameter_name [SCOPE = { MEMORY | SPFILE | BOTH }] SID = 'sid'
alter_system_set_clause	parameter_name = parameter_value [, parameter_value]... [COMMENT 'text'] [DEFERRED]

Subclause	Syntax
	[SCOPE = { MEMORY | SPFILE | BOTH }] [SID = { 'sid' | * }]
alter_table_partitioning	{ modify_table_default_attrs | set_subpartition_template | modify_table_partition | modify_table_subpartition | move_table_partition | move_table_subpartition | add_table_partition | coalesce_table_partition | drop_table_partition | drop_table_subpartition | rename_partition_subpart | truncate_partition_subpart | split_table_partition | split_table_subpartition | merge_table_partitions | merge_table_subpartitions | exchange_partition_subpart }
alter_table_properties	{ { physical_attributes_clause | logging_clause | table_compression | supplemental_table_logging | allocate_extent_clause | deallocate_unused_clause | shrink_clause | { CACHE | NOCACHE } | upgrade_table_clause | records_per_block_clause | parallel_clause | row_movement_clause } [physical_attributes_clause | logging_clause | table_compression | supplemental_table_logging | allocate_extent_clause | deallocate_unused_clause | shrink_clause | { CACHE | NOCACHE } | upgrade_table_clause | records_per_block_clause | parallel_clause | row_movement_clause]... | RENAME TO new_table_name } [alter_iot_clauses]

Subclause	Syntax
alter_tempfile_clause	TEMPFILE { 'filename' [, 'filename']... | filenumber [, filenumber]... } { RESIZE size_clause | autoextend_clause | DROP [INCLUDING DATAFILES] | ONLINE | OFFLINE }
alter_varray_col_properties	MODIFY VARRAY varray_item (modify_LOB_parameters)
analytic_clause	[query_partition_clause] [order_by_clause [windowing_clause]]
archive_log_clause	ARCHIVE LOG [INSTANCE 'instance_name' | THREAD integer] { { SEQUENCE integer | CHANGE integer | CURRENT [NOSWITCH] | GROUP integer | LOGFILE 'filename' [USING BACKUP CONTROLFILE] | NEXT | ALL | START } [TO 'location'] | STOP }
array_DML_clause	[WITH | WITHOUT] ARRAY DML [([schema.]type [, [schema.]varray_type]) [, ([schema.]type [, [schema.]varray_type])...]
attribute_clause	ATTRIBUTE level DETERMINES { dependent_column | (dependent_column [, dependent_column]...) }
auditing_by_clause	BY { proxy [, proxy]... | user [, user]... }

Subclause	Syntax
auditing_on_clause	ON { [schema.]object | DIRECTORY directory_name | DEFAULT }
autoextend_clause	AUTOEXTEND { OFF | ON [NEXT size_clause] [maxsize_clause] }
binding_clause	BINDING (parameter_type [, parameter_type]...) RETURN return_type [implementation_clause] using_function_clause [, (parameter_type [, parameter_type]...) RETURN return_type [implementation_clause] using_function_clause]...
bitmap_join_index_clause	[schema.]table ([[schema.]table. | t_alias.]column [ASC | DESC] [, [[schema.]table. | t_alias.]column [ASC | DESC]]...) FROM [schema.]table [t_alias] [, [schema.]table [t_alias]]... WHERE condition [local_partitioned_index] index_attributes
build_clause	BUILD { IMMEDIATE | DEFERRED }
C_declaration	C [NAME name] LIBRARY lib_name [AGENT IN (argument[, argument]...)] [WITH CONTEXT] [PARAMETERS (parameter[, parameter]...)]
call_spec	LANGUAGE { Java_declaration | C_declaration }
cancel_clause	CANCEL [IMMEDIATE] [WAIT | NOWAIT]

Subclause	Syntax
cell_assignment	measure_column [{ { condition | expr | single_column_for_loop } [, { condition | expr | single_column_for_loop }]... | multi_column_for_loop }] Note: The outer square brackets are part of the syntax. In this case, they do not indicate optionality.
cell_reference_options	[{ IGNORE | KEEP } NAV] [UNIQUE { DIMENSION | SINGLE REFERENCE }]
character_set_clause	CHARACTER SET character_set
check_datafiles_clause	CHECK DATAFILES [GLOBAL | LOCAL]
check_diskgroup_clauses	CHECK { ALL | DISK disk_name [, disk_name]... | DISKS IN FAILGROUP failgroup_name [, failgroup_name]... | FILE filename [, filename]... } [CHECK { ALL | DISK disk_name [, disk_name]... | DISKS IN FAILGROUP failgroup_name [, failgroup_name]... | FILE filename [, filename]... }]... [REPAIR | NOREPAIR]

Subclause	Syntax
checkpoint_clause	CHECKPOINT [GLOBAL | LOCAL]
cluster_index_clause	CLUSTER [schema.] cluster index_attributes
coalesce_index_partition	COALESCE PARTITION [parallel_clause]
coalesce_table_partition	COALESCE PARTITION [update_index_clauses] [parallel_clause]
column_association	COLUMNS [schema.]table.column [, [schema.]table.column]... using_statistics_type
column_clauses	{ { add_column_clause | modify_column_clause | drop_column_clause } [add_column_clause | modify_column_clause | drop_column_clause]... | rename_column_clause | modify_collection_retrieval [modify_collection_retrieval]... | modify_LOB_storage_clause | alter_varray_col_properties }
column_properties	{ object_type_col_properties | nested_table_col_properties | { varray_col_properties | LOB_storage_clause } [(LOB_partition_storage [, LOB_partition_storage]...)] | XMLType_column_properties } [{ object_type_col_properties | nested_table_col_properties | { varray_col_properties | LOB_storage_clause } [(LOB_partition_storage [, LOB_partition_storage]...)] | XMLType_column_properties }]...

Subclause	Syntax
commit_switchover_clause	{ PREPARE | COMMIT } TO SWITCHOVER [TO { { PHYSICAL | LOGICAL } PRIMARY | PHYSICAL STANDBY [{ WITH | WITHOUT } SESSION SHUTDOWN { WAIT | NOWAIT }] | LOGICAL STANDBY } | CANCEL]
compile_type_clause	COMPILE [DEBUG] [SPECIFICATION | BODY] [compiler_parameters_clause [compiler_parameters_clause] ...] [REUSE SETTINGS]
compiler_parameters_clause	parameter_name = parameter_value
composite_partitioning	PARTITION BY RANGE (column_list) [subpartition_by_list | subpartition_by_hash] (PARTITION [partition] range_values_clause table_partition_description [, PARTITION [partition] range_values_clause table_partition_description] ...)
compute_statistics_clause	COMPUTE [SYSTEM] STATISTICS [for_clause]
conditional_insert_clause	[ALL | FIRST] WHEN condition THEN insert_into_clause [values_clause] [error_logging_clause] [insert_into_clause [values_clause] [error_logging_clause]]... [WHEN condition THEN insert_into_clause [values_clause] [error_logging_clause] [insert_into_clause [values_clause] [error_logging_clause]]...]... [ELSE insert_into_clause

Subclause	Syntax
	[values_clause] [error_logging_clause] [insert_into_clause [values_clause] [error_logging_clause]]...]
constraint	{ inline_constraint | out_of_line_constraint | inline_ref_constraint | out_of_line_ref_constraint }
constraint_clauses	{ ADD { out_of_line_constraint [out_of_line_constraint]... | out_of_line_REF_constraint } | MODIFY { CONSTRAINT constraint | PRIMARY KEY | UNIQUE (column [, column]...) } constraint_state | RENAME CONSTRAINT old_name TO new_name | drop_constraint_clause }
constraint_state	[[[NOT] DEFERRABLE] [INITIALLY { IMMEDIATE | DEFERRED }] | [INITIALLY { IMMEDIATE | DEFERRED }] [[NOT] DEFERRABLE]] [RELY | NORELY] [using_index_clause] [ENABLE | DISABLE] [VALIDATE | NOVALIDATE] [exceptions_clause]
constructor_declaration	[FINAL] [INSTANTIABLE] CONSTRUCTOR FUNCTION datatype [[SELF IN OUT datatype,] parameter datatype [, parameter datatype]...] RETURN SELF AS RESULT { IS | AS } { pl/sql_block | call_spec }
constructor_spec	[FINAL] [INSTANTIABLE] CONSTRUCTOR FUNCTION datatype [([SELF IN OUT datatype,]

Subclause	Syntax
	parameter datatype [, parameter datatype]...)] RETURN SELF AS RESULT [{ IS | AS } call_spec]
context_clause	[WITH INDEX CONTEXT, SCAN CONTEXT implementation_type [COMPUTE ANCILLARY DATA]] [WITH COLUMN CONTEXT]
controlfile_clauses	{ CREATE [LOGICAL | PHYSICAL] STANDBY CONTROLFILE AS 'filename' [REUSE] | BACKUP CONTROLFILE TO { 'filename' [REUSE] | trace_file_clause } }
create_datafile_clause	CREATE DATAFILE { 'filename' | filenumber } [, 'filename' | filenumber]... } [AS { file_specification [, file_specification]... | NEW }]
create_incomplete_type	CREATE [OR REPLACE] TYPE [schema.]type_name ;
create_mv_refresh	{ REFRESH { { FAST | COMPLETE | FORCE } | ON { DEMAND | COMMIT } | { START WITH | NEXT } date | WITH { PRIMARY KEY | ROWID } | USING { DEFAULT [MASTER | LOCAL] ROLLBACK SEGMENT | [MASTER | LOCAL] ROLLBACK SEGMENT rollback_segment } [DEFAULT [MASTER | LOCAL] ROLLBACK SEGMENT | [MASTER | LOCAL] ROLLBACK SEGMENT rollback_segment]...

Subclause	Syntax
		USING { ENFORCED	TRUSTED } CONSTRAINTS } [{ FAST	COMPLETE	FORCE } | ON { DEMAND	COMMIT } | { START WITH	NEXT } date | WITH { PRIMARY KEY	ROWID } | USING { DEFAULT [MASTER	LOCAL] ROLLBACK SEGMENT | [MASTER	LOCAL] ROLLBACK SEGMENT rollback_segment } [DEFAULT [MASTER	LOCAL] ROLLBACK SEGMENT | [MASTER	LOCAL] ROLLBACK SEGMENT rollback_segment]... | USING { ENFORCED	TRUSTED } CONSTRAINTS]... | NEVER REFRESH }
create_nested_table_type	CREATE [OR REPLACE] TYPE [schema.]type_name [OID 'object_identifier'] { IS	AS } TABLE OF datatype ;
create_object_type	CREATE [OR REPLACE] TYPE [schema.]type_name [OID 'object_identifier'] [invoker_rights_clause] { { IS	AS } OBJECT 	UNDER [schema.]supertype } [sqlj_object_type] [(attribute datatype [sqlj_object_type_attr] [, attribute datatype [sqlj_object_type_attr]... [, element_spec [, element_spec]...])] [[NOT] FINAL] [[NOT] INSTANTIABLE] ;

Subclause	Syntax
create_varray_type	CREATE [OR REPLACE] TYPE [schema.]type_name [OID 'object_identifier'] { IS | AS } { VARRAY | VARYING ARRAY } (limit) OF datatype ;
database_file_clauses	{ RENAME FILE 'filename' [, 'filename']... TO 'filename' | create_datafile_clause | alter_datafile_clause | alter_tempfile_clause }
database_logging_clauses	{ LOGFILE [GROUP integer] file_specification [, [GROUP integer] file_specification]... | MAXLOGFILES integer | MAXLOGMEMBERS integer | MAXLOGHISTORY integer | { ARCHIVELOG | NOARCHIVELOG } | FORCE LOGGING }
datafile_tempfile_clauses	{ ADD { DATAFILE | TEMPFILE } [file_specification [, file_specification]...] | RENAME DATAFILE 'filename' [, 'filename']... TO 'filename' [, 'filename']... | { DATAFILE | TEMPFILE } { ONLINE | OFFLINE } }
datafile_tempfile_spec	['filename'] [SIZE size_clause] [REUSE] [autoextend_clause]
dblink	database[.domain [.domain]...] [@ connect_descriptor]
dblink_authentication	AUTHENTICATED BY user IDENTIFIED BY password
deallocate_unused_clause	DEALLOCATE UNUSED [KEEP size_clause]

Subclause	Syntax
default_cost_clause	DEFAULT COST (cpu_cost, io_cost, network_cost)
default_selectivity_clause	DEFAULT SELECTIVITY default_selectivity
default_tablespace	DEFAULT TABLESPACE tablespace [DATAFILE datafile_tempfile_spec] extent_management_clause
default_settings_clauses	{ SET DEFAULT { BIGFILE | SMALLFILE } TABLESPACE | DEFAULT TABLESPACE tablespace | DEFAULT TEMPORARY TABLESPACE { tablespace | tablespace_group_name } | RENAME GLOBAL_NAME TO database.domain [.domain]... | { ENABLE BLOCK CHANGE TRACKING [USING FILE 'filename' [REUSE]] | DISABLE BLOCK CHANGE TRACKING } | flashback_mode_clause | set_time_zone_clause }
default_temp_tablespace	[BIGFILE | SMALLFILE] DEFAULT TEMPORARY TABLESPACE tablespace [TEMPFILE file_specification [, file_specification]...] extent_management_clause
dependent_handling_clause	{ INVALIDATE | CASCADE [{ [NOT] INCLUDING TABLE DATA | CONVERT TO SUBSTITUTABLE }] [[FORCE] exceptions_clause] }
dimension_join_clause	JOIN KEY { child_key_column | (child_key_column [, child_key_column]...) } REFERENCES parent_level [JOIN KEY { child_key_column | (child_key_column [, child_key_column]...) } REFERENCES parent_level]...

Subclause	Syntax
disk_clauses	{ diskgroup_name { add_disk_clause | drop_disk_clauses | resize_disk_clauses } | { diskgroup_name | ALL } undrop_disk_clause }
diskgroup_alias_clauses	{ ADD ALIAS alias_name FOR filename [, alias_name FOR filename]... | DROP ALIAS alias_name [, alias_name]... | RENAME ALIAS old_alias_name TO new_alias_name [, old_alias_name TO new_alias_name]... }
diskgroup_availability	{ MOUNT | DISMOUNT [FORCE | NOFORCE] }
diskgroup_clauses	{ diskgroup_name { rebalance_diskgroup_clause | check_diskgroup_clauses | diskgroup_template_clauses | diskgroup_directory_clauses | diskgroup_alias_clauses | drop_diskgroup_file_clause } | { diskgroup_name | ALL } diskgroup_availability }
diskgroup_directory_clauses	{ ADD DIRECTORY filename [, filename]... | DROP DIRECTORY filename [FORCE | NOFORCE] [, filename [FORCE | NOFORCE]]... | RENAME DIRECTORY old_dir_name TO new_dir_name [, old_dir_name TO new_dir_name]... }
diskgroup_file_spec	[' { fully_qualified_file_name | numeric_file_name | incorporate_file_name | alias_file_name }

Subclause	Syntax
	'] [SIZE size_clause] [REUSE] [autoextend_clause]
diskgroup_template_clauses	{ { ADD | ALTER } TEMPLATE qualified_template_clause [, qualified_template_clause]... | DROP TEMPLATE template_name [, template_name]... }
distributed_recov_clauses	{ ENABLE | DISABLE } DISTRIBUTED RECOVERY
dml_event_clause	{ DELETE | INSERT | UPDATE [OF column [, column]...] } [OR { DELETE | INSERT | UPDATE [OF column [, column]...] }]... ON { [schema.]table | [NESTED TABLE nested_table_column OF] [schema.] view } [referencing_clause] [FOR EACH ROW]
dml_table_expression_clause	{ [schema.] { table [{ PARTITION (partition) | SUBPARTITION (subpartition) } | @ dblink] | { view | materialized view } [@ dblink] } | (subquery [subquery_restriction_clause]) | table_collection_expression }
domain_index_clause	INDEXTYPE IS indextype [parallel_clause] [PARAMETERS ('ODCI_parameters')]
drop_binding_clause	DROP BINDING (parameter_type [, parameter_type]...) [FORCE]

Subclause	Syntax
drop_column_clause	{ SET UNUSED { COLUMN column | (column [, column]...) } [{ CASCADE CONSTRAINTS | INVALIDATE } [CASCADE CONSTRAINTS | INVALIDATE]...] | DROP { COLUMN column | (column [, column]...) } [{ CASCADE CONSTRAINTS | INVALIDATE } [CASCADE CONSTRAINTS | INVALIDATE]...] [CHECKPOINT integer] | DROP { UNUSED COLUMNS | COLUMNS CONTINUE } [CHECKPOINT integer] }
drop_constraint_clause	DROP { { PRIMARY KEY | UNIQUE (column [, column]...) } [CASCADE] [{ KEEP | DROP } INDEX] | CONSTRAINT constraint [CASCADE] }
drop_disk_clauses	DROP { DISK disk_name [FORCE | NOFORCE] [, disk_name [FORCE | NOFORCE]]... | DISKS IN FAILGROUP failgroup_name [FORCE | NOFORCE] [, failgroup_name [FORCE | NOFORCE]]... }
drop_diskgroup_file_clause	DROP FILE filename [, filename]...
drop_index_partition	DROP PARTITION partition_name
drop_logfile_clauses	DROP [STANDBY] LOGFILE { logfile_descriptor [, logfile_descriptor]... | MEMBER 'filename' [, 'filename']... }

Subclause	Syntax
drop_table_partition	DROP PARTITION partition [update_index_clauses [parallel_clause]]
drop_table_subpartition	DROP SUBPARTITION subpartition [update_index_clauses [parallel_clause]]
element_spec	[inheritance_clauses] { subprogram_spec | constructor_spec | map_order_function_spec } [subprogram_clause | constructor_spec | map_order_function_spec]... [, pragma_clause]
else_clause	ELSE else_expr
enable_disable_clause	{ ENABLE | DISABLE } [VALIDATE | NOVALIDATE] { UNIQUE (column [, column]...) | PRIMARY KEY | CONSTRAINT constraint } [using_index_clause] [exceptions_clause] [CASCADE] [{ KEEP | DROP } INDEX]
end_session_clauses	{ DISCONNECT SESSION 'integer1, integer2' [POST_TRANSACTION] | KILL SESSION 'integer1, integer2' } [IMMEDIATE]
estimate_statistics_clause	ESTIMATE [SYSTEM] STATISTICS [for_clause] [SAMPLE integer { ROWS | PERCENT }]
exceptions_clause	EXCEPTIONS INTO [schema.]table
exchange_partition_subpart	EXCHANGE { PARTITION partition | SUBPARTITION subpartition } WITH TABLE table [{ INCLUDING | EXCLUDING } INDEXES] [{ WITH | WITHOUT } VALIDATION] [exceptions_clause] [update_index_clauses [parallel_clause]]

Subclause	Syntax
expr	{ simple_expression | compound_expression | case_expression | cursor_expression | datetime_expression | function_expression | interval_expression | object_access_expression | scalar_subquery_expression | model_expression | type_constructor_expression | variable_expression }
expression_list	{ expr [, expr]... | (expr [, expr]...) }
extended_attribute_clause	ATTRIBUTE attribute LEVEL level DETERMINES { dependent_column | (dependent_column [, dependent_column]...) [LEVEL level DETERMINES { dependent_column | (dependent_column [, dependent_column]...)]...
extent_management_clause	EXTENT MANAGEMENT { DICTIONARY | LOCAL [AUTOALLOCATE | UNIFORM [SIZE size_clause]] }
external_data_properties	DEFAULT DIRECTORY directory [ACCESS PARAMETERS { (opaque_format_spec) | USING CLOB subquery }] LOCATION ([directory:] 'location_specifier' [, [directory:] 'location_specifier']...)

Subclause	Syntax
external_table_clause	([TYPE access_driver_type] external_data_properties) [REJECT LIMIT { integer | UNLIMITED }]
file_specification	{ datafile_tempfile_spec | diskgroup_file_spec | redo_log_file_spec }
finish_clause	[DISCONNECT [FROM SESSION]] [parallel_clause] FINISH [SKIP [STANDBY LOGFILE]] [WAIT | NOWAIT]
flashback_mode_clause	FLASHBACK { ON | OFF }
flashback_query_clause	[VERSIONS BETWEEN { SCN | TIMESTAMP } { expr | MINVALUE } AND { expr | MAXVALUE }] AS OF { SCN | TIMESTAMP } expr
for_clause	FOR { TABLE | ALL [INDEXED] COLUMNS [SIZE integer] | COLUMNS [SIZE integer] { column | attribute } [SIZE integer] [{ column | attribute } [SIZE integer]]... | ALL [LOCAL] INDEXES } [FOR { TABLE | ALL [INDEXED] COLUMNS [SIZE integer] | COLUMNS [SIZE integer] { column | attribute } [SIZE integer] [{ column | attribute } [SIZE integer]]... | ALL [LOCAL] INDEXES }]...
for_update_clause	FOR UPDATE [OF [[schema.] { table | view } .]column [, [[schema.]

Subclause	Syntax
	{ table | view } .]column]...] [NOWAIT | WAIT integer]
full_database_recovery	[STANDBY] DATABASE [{ UNTIL { CANCEL | TIME date | CHANGE integer } | USING BACKUP CONTROLFILE } [UNTIL { CANCEL | TIME date | CHANGE integer } | USING BACKUP CONTROLFILE]...]
fully_qualified_file_name	+diskgroup_name/db_name/file_type/ file_type_tag.filenumber.incarnation_number
function_association	{ FUNCTIONS [schema.]function [, [schema.]function]... | PACKAGES [schema.]package [, [schema.]package]... | TYPES [schema.]type [, [schema.]type]... | INDEXES [schema.]index [, [schema.]index]... | INDEXTYPES [schema.]indextype [, [schema.]indextype]... } { using_statistics_type | { default_cost_clause [, default_selectivity_clause] | default_selectivity_clause [, default_cost_clause] } }
function_declaration	FUNCTION name (parameter datatype[, parameter datatype]...) RETURN datatype { IS | AS } { pl/sql_block | call_spec }

Subclause	Syntax
function_spec	FUNCTION name (parameter datatype [, parameter datatype]...) return_clause
general_recovery	RECOVER [AUTOMATIC] [FROM 'location'] { { full_database_recovery | partial_database_recovery | LOGFILE 'filename' } [{ TEST | ALLOW integer CORRUPTION | parallel_clause } [TEST | ALLOW integer CORRUPTION | parallel_clause]...] | CONTINUE [DEFAULT] | CANCEL }
global_partitioned_index	GLOBAL PARTITION BY { RANGE (column_list) (index_partitioning_clause) | HASH (column_list) { individual_hash_partitions | hash_partitions_by_quantity } }
grant_object_privileges	{ object_privilege | ALL [PRIVILEGES] } [(column [, column]...)] [, { object_privilege | ALL [PRIVILEGES] } [(column [, column]...)]]... on_object_clause TO grantee_clause [WITH HIERARCHY OPTION] [WITH GRANT OPTION]
grant_system_privileges	{ system_privilege | role | ALL PRIVILEGES } [, { system_privilege

Subclause	Syntax
		role | ALL PRIVILEGES }]... TO grantee_clause [IDENTIFIED BY password] [WITH ADMIN OPTION]
grantee_clause	{ user | role | PUBLIC } [, { user | role | PUBLIC }]...
group_by_clause	GROUP BY { expr | rollup_cube_clause | grouping_sets_clause } [, { expr | rollup_cube_clause | grouping_sets_clause }]... [HAVING condition]
grouping_expression_list	expression_list [, expression_list]...
grouping_sets_clause	GROUPING SETS ({ rollup_cube_clause | grouping_expression_list })
hash_partitioning	PARTITION BY HASH (column [, column] ...) { individual_hash_partitions | hash_partitions_by_quantity }
hash_partitions_by_quantity	PARTITIONS hash_partition_quantity [STORE IN (tablespace [, tablespace]...)] [OVERFLOW STORE IN (tablespace [, tablespace]...)]
hierarchical_query_clause	[START WITH condition] CONNECT BY [NOCYCLE] condition
hierarchy_clause	HIERARCHY hierarchy (child_level CHILD OF parent_level [CHILD OF parent_level]... [dimension_join_clause])

Subclause	Syntax
implementation_clause	{ ANCILLARY TO primary_operator (parameter_type [, parameter_type]...) [, primary_operator (parameter_type [, parameter_type]...)]... | context_clause }
incomplete_file_name	+diskgroup_name [(template_name)]
index_attributes	[{ physical_attributes_clause | logging_clause | ONLINE | COMPUTE STATISTICS | TABLESPACE { tablespace | DEFAULT } | key_compression | { SORT | NOSORT } | REVERSE | parallel_clause } [physical_attributes_clause | logging_clause | ONLINE | COMPUTE STATISTICS | TABLESPACE { tablespace | DEFAULT } | key_compression | { SORT | NOSORT } | REVERSE | parallel_clause]...]
index_expr	{ column | column_expression }
index_org_overflow_clause	[INCLUDING column_name] OVERFLOW [segment_attributes_clause]
index_org_table_clause	[{ mapping_table_clause | PCTTHRESHOLD integer | key_compression } [mapping_table_clause | PCTTHRESHOLD integer | key_compression]...] [index_org_overflow_clause]

Subclause	Syntax
index_partition_description	PARTITION [partition [{ segment_attributes_clause | key_compression } [segment_attributes_clause | key_compression]...]]
index_partitioning_clause	PARTITION [partition] VALUES LESS THAN (value[, value...]) [segment_attributes_clause]
index_properties	[{ { global_partitioned_index | local_partitioned_index } | index_attributes } [{ { global_partitioned_index | local_partitioned_index } | index_attributes }]... | domain_index_clause]
index_subpartition_clause	{ STORE IN (tablespace[, tablespace]...) | (SUBPARTITION [subpartition [TABLESPACE tablespace]] [, SUBPARTITION [subpartition [TABLESPACE tablespace]]]...) }
individual_hash_partitions	(PARTITION [partition partitioning_storage_clause] [, PARTITION [partition partitioning_storage_clause]]...)
inheritance_clauses	[NOT] { OVERRIDING | FINAL | INSTANTIABLE } [[NOT] { OVERRIDING | FINAL | INSTANTIABLE }]...

Subclause	Syntax
inline_constraint	[CONSTRAINT constraint_name] { [NOT] NULL | UNIQUE | PRIMARY KEY | references_clause | CHECK (condition) } [constraint_state]
inline_ref_constraint	{ SCOPE IS [schema.] scope_table | WITH ROWID | [CONSTRAINT constraint_name] references_clause [constraint_state] }
inner_cross_join_clause	table_reference { [INNER] JOIN table_reference { ON condition | USING (column [, column]...) } | { CROSS | NATURAL [INNER] } JOIN table_reference }
insert_into_clause	INTO dml_table_expression_clause [t_alias] [(column [, column]...)]
integer	[+ | -] digit [digit]...
interval_day_to_second	INTERVAL '{ integer | integer time_expr | time_expr }' { { DAY | HOUR | MINUTE } [(leading_precision)] | SECOND [(leading_precision [, fractional_seconds_precision])] } [TO { DAY | HOUR | MINUTE | SECOND [(fractional_seconds_precision)] }]
interval_year_to_month	INTERVAL 'integer [- integer]' { YEAR | MONTH } [(precision)] [TO { YEAR | MONTH }]

Subclause	Syntax
into_clause	INTO [schema.] table
invoker_rights_clause	AUTHID { CURRENT_USER | DEFINER }
Java_declaration	JAVA NAME 'string'
join_clause	{ inner_cross_join_clause | outer_join_clause }
key_compression	{ COMPRESS [integer] | NOCOMPRESS }
level_clause	LEVEL level IS { level_table.level_column | (level_table.level_column [, level_table.level_column]...) }
list_partitioning	PARTITION BY LIST (column) (PARTITION [partition] list_values_clause table_partition_description [, PARTITION [partition] list_values_clause table_partition_description]...)
list_values_clause	VALUES ({ value | NULL [, { value | NULL }...) | DEFAULT)
LOB_parameters	{ TABLESPACE tablespace | { ENABLE | DISABLE } STORAGE IN ROW | storage_clause | CHUNK integer | PCTVERSION integer | RETENTION | FREEPOOLS integer | { CACHE | { NOCACHE | CACHE READS } [logging_clause] } } [TABLESPACE tablespace | { ENABLE | DISABLE } STORAGE IN ROW | storage_clause

Subclause	Syntax
		CHUNK integer | PCTVERSION integer | RETENTION | FREEPOOLS integer | { CACHE | { NOCACHE	CACHE READS } [logging_clause] }]...
LOB_partition_storage	PARTITION partition { LOB_storage_clause	varray_col_properties } [LOB_storage_clause	varray_col_properties]... [(SUBPARTITION subpartition { LOB_storage_clause | varray_col_properties } [LOB_storage_clause | varray_col_properties]...)]
LOB_storage_clause	LOB { (LOB_item [, LOB_item]...) STORE AS (LOB_parameters) | (LOB_item) STORE AS { LOB_segname (LOB_parameters) | LOB_segname | (LOB_parameters) } }
local_partitioned_index	LOCAL [on_range_partitioned_table | on_list_partitioned_table | on_hash_partitioned_table | on_comp_partitioned_table]
logfile_clause	LOGFILE [GROUP integer] file_specification [, [GROUP integer] file_specification]...
logfile_clauses	{ { ARCHIVELOG [MANUAL] | NOARCHIVELOG } | [NO] FORCE LOGGING | RENAME FILE 'filename' [, 'filename']... TO 'filename'

Subclause	Syntax
	| CLEAR [UNARCHIVED] LOGFILE logfile_descriptor [, logfile_descriptor]... [UNRECOVERABLE DATAFILE] | add_logfile_clauses | drop_logfile_clauses | supplemental_db_logging }
logfile_descriptor	{ GROUP integer | ('filename' [, 'filename']...) | 'filename' }
logging_clause	{ LOGGING | NOLOGGING }
main_model	[MAIN main_model_name] model_column_clauses [cell_reference_options] model_rules_clause
managed_standby_recovery	RECOVER MANAGED STANDBY DATABASE [recover_clause | cancel_clause | finish_clause]
map_order_func_declaration	{ MAP | ORDER } MEMBER function_declaration
map_order_function_spec	{ MAP | ORDER } MEMBER function_spec
mapping_table_clauses	{ MAPPING TABLE | NOMAPPING }
materialized_view_props	[column_properties] [table_partitioning_clauses] [CACHE | NOCACHE] [parallel_clause] [build_clause]
maximize_standby_db_clause	SET STANDBY DATABASE TO MAXIMIZE { PROTECTION | AVAILABILITY | PERFORMANCE }
maxsize_clause	MAXSIZE { UNLIMITED | size_clause }
merge_insert_clause	WHEN NOT MATCHED THEN INSERT [(column [, column]...)] VALUES ({ expr [, expr]... | DEFAULT }) [where_clause]

Subclause	Syntax
merge_table_partitions	MERGE PARTITIONS partition_1, partition_2 [INTO partition_spec] [update_index_clauses] [parallel_clause]
merge_table_subpartitions	MERGE SUBPARTITIONS subpart_1, subpart_2 [INTO subpartition_spec] [update_index_clauses] [parallel_clause]
merge_update_clause	WHEN MATCHED THEN UPDATE SET column = { expr | DEFAULT } [, column = { expr | DEFAULT }]... [where_clause] [DELETE where_clause]
model_clause	MODEL [cell_reference_options] [return_rows_clause] [reference_model] [reference_model]... main_model
model_column	expr [[AS] c_alias]
model_column_clauses	[query_partition_clause [c_alias]] DIMENSION BY (model_column [, model_column]...) MEASURES (model_column [, model_column]...)
model_rules_clause	RULES [UPSERT | UPDATE] [{ AUTOMATIC | SEQUENTIAL } ORDER] [ITERATE (number) [UNTIL (condition)]] ([UPDATE | UPSERT] cell_assignment [order_by_clause] = expr [[UPDATE | UPSERT] cell_assignment [order_by_clause] = expr]...)
modify_col_properties	(column [datatype] [DEFAULT expr] [inline_constraint [inline_constraint]...] [LOB_storage_clause] [, column [datatype] [DEFAULT expr] [inline_constraint

Subclause	Syntax
	[inline_constraint]...] [LOB_storage_clause]])
modify_col_substitutable	COLUMN column [NOT] SUBSTITUTABLE AT ALL LEVELS [FORCE]
modify_collection_retrieval	MODIFY NESTED TABLE collection_item RETURN AS { LOCATOR | VALUE }
modify_column_clauses	MODIFY { modify_col_properties | modify_col_substitutable }
modify_hash_partition	MODIFY PARTITION partition { partition_attributes | alter_mapping_table_clause | [REBUILD] UNUSABLE LOCAL INDEXES }
modify_hash_subpartition	{ { allocate_extent_clause | deallocate_unused_clause | shrink_clause | { LOB LOB_item | VARRAY varray } modify_LOB_parameters [{ LOB LOB_item | VARRAY varray } modify_LOB_parameters]... } | [REBUILD] UNUSABLE LOCAL INDEXES }
modify_index_default_attrs	MODIFY DEFAULT ATTRIBUTES [FOR PARTITION partition] { physical_attributes_clause | TABLESPACE { tablespace | DEFAULT } | logging_clause } [physical_attributes_clause | TABLESPACE { tablespace | DEFAULT } | logging_clause]...

Subclause	Syntax

modify_index_partition	MODIFY PARTITION partition
	{ { deallocate_unused_clause
	  | allocate_extent_clause
	  | physical_attributes_clause
	  | logging_clause
	  | key_compression
	  }
	  [deallocate_unused_clause
	  | allocate_extent_clause
	  | physical_attributes_clause
	  | logging_clause
	  | key_compression
	  ]...
	| PARAMETERS ('ODCI_parameters')
	| COALESCE
	| UPDATE BLOCK REFERENCES
	| UNUSABLE
	}

modify_index_subpartition	MODIFY SUBPARTITION subpartition { UNUSABLE | allocate_extent_clause | deallocate_unused_clause }

modify_list_partition	MODIFY PARTITION partition
	  { partition_attributes
	  | {ADD | DROP} VALUES
	  (partition_value[, partition_value]...)
	  | [REBUILD] UNUSABLE LOCAL INDEXES
	  }

modify_list_subpartition	{ allocate_extent_clause
	| deallocate_unused_clause
	| shrink_clause
	| { LOB LOB_item | VARRAY varray }
	  modify_LOB_parameters
	  [{ LOB LOB_item | VARRAY varray }
	  modify_LOB_parameters
	  ] ...
	| [REBUILD] UNUSABLE LOCAL INDEXES
	| { ADD | DROP } VALUES (value[, value]...)
	}

modify_LOB_parameters	{ storage_clause | PCTVERSION integer | RETENTION | FREEPOOLS integer | REBUILD FREEPOOLS | { CACHE | { NOCACHE | CACHE READS } [logging_clause]

Subclause	Syntax
	} | allocate_extent_clause | deallocate_unused_clause | shrink_clause } [storage_clause | PCTVERSION integer | RETENTION | FREEPOOLS integer | REBUILD FREEPOOLS | { CACHE | { NOCACHE | CACHE READS } [logging_clause] } | allocate_extent_clause | deallocate_unused_clause | shrink_clause]...
modify_LOB_storage_clause	MODIFY LOB (LOB_item) (modify_LOB_parameters)
modify_range_partition	MODIFY PARTITION partition { partition_attributes | { add_hash_subpartition | add_list_subpartition } | COALESCE SUBPARTITION [update_index_clauses] [parallel_clause] | alter_mapping_table_clause | [REBUILD] UNUSABLE LOCAL INDEXES }
modify_table_default_attrs	MODIFY DEFAULT ATTRIBUTES [FOR PARTITION partition] [segment_attributes_clause] [table_compression] [PCTTHRESHOLD integer] [key_compression] [alter_overflow_clause] [{ LOB (LOB_item) | VARRAY varray } (LOB_parameters) [{ LOB (LOB_item) | VARRAY varray } (LOB_parameters)]...]

Subclause	Syntax
modify_table_partition	{ modify_range_partition | modify_hash_partition | modify_list_partition }
modify_table_subpartition	MODIFY SUBPARTITION subpartition { modify_hash_subpartition | modify_list_subpartition }
move_table_clause	MOVE [ONLINE] [segment_attributes_clause] [table_compression] [index_org_table_clause] [{ LOB_storage_clause | varray_col_properties } [{ LOB_storage_clause | varray_col_properties }]...] [parallel_clause]
move_table_partition	MOVE PARTITION partition [MAPPING TABLE] [table_partition_description] [update_index_clauses] [parallel_clause]
move_table_subpartition	MOVE SUBPARTITION subpartition_spec [update_index_clauses] [parallel_clause]

multi_column_for_loop	FOR (dimension_column
	[, dimension_column]...)
	IN ({ (literal [, literal]...)
	[(literal [, literal]...)...]
	| subquery
	}
	)

multi_table_insert	{ ALL insert_into_clause [values_clause] [insert_into_clause [values_clause]]... | conditional_insert_clause } subquery

Subclause	Syntax
multiset_except	nested_table1 MULTISET EXCEPT [ALL | DISTINCT] nested_table2
multiset_intersect	nested_table1 MULTISET INTERSECT [ALL | DISTINCT] nested_table2
multiset_union	nested_table1 MULTISET UNION [ALL | DISTINCT] nested_table2
nested_table_col_properties	NESTED TABLE { nested_item | COLUMN_VALUE } [substitutable_column_clause] STORE AS storage_table [({ (object_properties) | [physical_properties] | [column_properties] } [(object_properties) | [physical_properties] | [column_properties]]...)] [RETURN AS { LOCATOR | VALUE }]
new_values_clause	{ INCLUDING | EXCLUDING } NEW VALUES
number	[+ | -] { digit [digit]... [.] [digit [digit]...] | . digit [digit]... } [e [+ | -] digit [digit]...] [f | d]
numeric_file_name	+diskgroup_name.filenumber.incarnation_number
object_properties	{ { column | attribute } [DEFAULT expr] [inline_constraint [inline_constraint]... | inline_ref_constraint] | { out_of_line_constraint | out_of_line_ref_constraint | supplemental_logging_props } }

Subclause	Syntax
object_table	CREATE [GLOBAL TEMPORARY] TABLE [schema.]table OF [schema.]object_type [object_table_substitution] [(object_properties)] [ON COMMIT { DELETE | PRESERVE } ROWS] [OID_clause] [OID_index_clause] [physical_properties] [table_properties] ;
object_table_substitution	[NOT] SUBSTITUTABLE AT ALL LEVELS
object_type_col_properties	COLUMN column substitutable_column_clause
object_view_clause	OF [schema.]type_name { WITH OBJECT IDENTIFIER { DEFAULT | (attribute [, attribute]...) } | UNDER [schema.]superview } ({ out_of_line_constraint | attribute inline_constraint [inline_constraint]... } [, { out_of_line_constraint | attribute inline_constraint [inline_constraint]... }]...)
OID_clause	OBJECT IDENTIFIER IS { SYSTEM GENERATED | PRIMARY KEY }
OID_index_clause	OIDINDEX [index] ({ physical_attributes_clause | TABLESPACE tablespace } [physical_attributes_clause | TABLESPACE tablespace]...)
on_comp_partitioned_table	[STORE IN (tablespace [, tablespace]...)] (PARTITION [partition [{ segment_attribute_clause | key_compression

Subclause	Syntax
	} [segment_attribute_clause | key_compression]...] [index_subpartition_clause]] [, PARTITION [partition [{ segment_attribute_clause | key_compression } [segment_attribute_clause | key_compression]...] [index_subpartition_clause]]...])
on_hash_partitioned_table	{ STORE IN (tablespace[, tablespace]...) | (PARTITION [partition [TABLESPACE tablespace]] [, PARTITION [partition [TABLESPACE tablespace]]]...) }
on_list_partitioned_table	(PARTITION [partition [{ segment_attributes_clause | key_compression } [segment_attributes_clause | key_compression]...]] [, PARTITION [partition [{ segment_attributes_clause | key_compression } [segment_attributes_clause | key_compression]...]]]...)

Subclause	Syntax

on_object_clause	{ schema.object
	| { DIRECTORY directory_name
	  | JAVA { SOURCE | RESOURCE } [schema.]object
	  }
	}

on_range_partitioned_table	(PARTITION
	  [partition
	  [{ segment_attributes_clause
	  | key_compression
	  }
	  [segment_attributes_clause
	  | key_compression
	  ]...
	  ]
	  ]
	  [, PARTITION
	  [partition
	  [{ segment_attributes_clause
	  | key_compression
	  }
	  [segment_attributes_clause
	  | key_compression
	  ]...
	  ]
	  ]
	  ]...
	)

order_by_clause	ORDER [SIBLINGS] BY
	{ expr | position | c_alias }
	[ASC | DESC]
	[NULLS FIRST | NULLS LAST]
	  [, { expr | position | c_alias }
	  [ASC | DESC]
	  [NULLS FIRST | NULLS LAST]
	  ]...

out_of_line_constraint	[CONSTRAINT constraint_name] { UNIQUE (column [, column]...) | PRIMARY KEY (column [, column]...) | FOREIGN KEY (column [, column]...) references_clause | CHECK (condition) } [constraint_state]

out_of_line_ref_constraint	{ SCOPE FOR
	  ({ ref_col | ref_attr })
	  IS [schema.]scope_table
	| REF
	  ({ ref_col | ref_attr })
	  WITH ROWID

Subclause	Syntax
	| [CONSTRAINT constraint_name] FOREIGN KEY ({ ref_col | ref_attr }) references_clause [constraint_state] }
outer_join_clause	table_reference [query_partition_clause] { outer_join_type JOIN | NATURAL [outer_join_type] JOIN } table_reference [query_partition_clause] [ON condition | USING (column [, column]...)]
outer_join_type	{ FULL | LEFT | RIGHT } [OUTER]
parallel_clause	{ NOPARALLEL | PARALLEL [integer] }
parallel_enable_clause	PARALLEL_ENABLE [(PARTITION argument BY { ANY | { HASH | RANGE } (column [, column]...) }) [streaming_clause]]
partial_database_recovery	{ TABLESPACE tablespace [, tablespace]... | DATAFILE { 'filename' | filenumber } [, 'filename' | filenumber]... } | STANDBY { TABLESPACE tablespace [, tablespace]... | DATAFILE { 'filename' | filenumber } [, 'filename' | filenumber]... } } UNTIL [CONSISTENT WITH] CONTROLFILE }
partition_attributes	[{ physical_attributes_clause | logging_clause | allocate_extent_clause | deallocate_unused_clause | shrink_clause }

Subclause	Syntax
	[physical_attributes_clause | logging_clause | allocate_extent_clause | deallocate_unused_clause | shrink_clause]...] [OVERFLOW { physical_attributes_clause | logging_clause | allocate_extent_clause | deallocate_unused_clause } [physical_attributes_clause | logging_clause | allocate_extent_clause | deallocate_unused_clause]...] [table_compression] [{ LOB LOB_item | VARRAY varray } modify_LOB_parameters [{ LOB LOB_item | VARRAY varray } modify_LOB_parameters]...]
partition_extended_name	[schema.] { table | view } [PARTITION (partition) | SUBPARTITION (subpartition)]
partition_level_subpartition	{ SUBPARTITIONS hash_subpartition_quantity [STORE IN (tablespace[, tablespace]...)] | (subpartition_spec[, subpartition_spec]...) }
partition_spec	PARTITION [partition] [table_partition_description]
partitioning_storage_clause	[{ TABLESPACE tablespace | OVERFLOW [TABLESPACE tablespace] | LOB (LOB_item) STORE AS { LOB_segname [(TABLESPACE tablespace)] | (TABLESPACE tablespace) } | VARRAY varray_item STORE AS LOB LOB_segname } [{ TABLESPACE tablespace | OVERFLOW [TABLESPACE tablespace] | LOB (LOB_item) STORE AS { LOB_segname [(TABLESPACE tablespace)]

Subclause	Syntax
		(TABLESPACE tablespace) } | VARRAY varray_item STORE AS LOB LOB_segname }]...]
password_parameters	{ { FAILED_LOGIN_ATTEMPTS | PASSWORD_LIFE_TIME | PASSWORD_REUSE_TIME | PASSWORD_REUSE_MAX | PASSWORD_LOCK_TIME | PASSWORD_GRACE_TIME } { expr | UNLIMITED | DEFAULT } | PASSWORD_VERIFY_FUNCTION { function | NULL | DEFAULT } }
permanent_tablespace_clause	{ MINIMUM EXTENT integer [K | M] | BLOCKSIZE integer [K] | logging_clause | FORCE LOGGING | DEFAULT [table_compression] storage_clause | { ONLINE | OFFLINE } | extent_management_clause | segment_management_clause | flashback_mode_clause [MINIMUM EXTENT integer [K | M] | BLOCKSIZE integer [K] | logging_clause | FORCE LOGGING | DEFAULT [table_compression] storage_clause | { ONLINE | OFFLINE } | extent_management_clause | segment_management_clause | flashback_mode_clause]... }
physical_attributes_clause	[{ PCTFREE integer | PCTUSED integer | INITRANS integer | storage_clause } [PCTFREE integer | PCTUSED integer | INITRANS integer | storage_clause]...]

Subclause	Syntax

physical_properties	{ segment_attributes_clause
	  [table_compression]
	| ORGANIZATION
	  { HEAP
	  [segment_attributes_clause]
	  [table_compression]
	  | INDEX
	  [segment_attributes_clause]
	  index_org_table_clause
	  | EXTERNAL
	  external_table_clause
	  }
	| CLUSTER cluster (column [, column]...)
	}

pragma_clause	PRAGMA RESTRICT_REFERENCES ({ method_name | DEFAULT } , { RNDS | WNDS | RNPS | WNPS | TRUST } [, { RNDS | WNDS | RNPS | WNPS | TRUST }]...)
procedure_declaration	PROCEDURE name (parameter datatype [, parameter datatype]...) { IS | AS } { pl/sql_block | call_spec }
procedure_spec	PROCEDURE name (parameter datatype [, parameter datatype]...) [{ IS | AS } call_spec]
proxy_authentication	{ AUTHENTICATION REQUIRED | AUTHENTICATED USING { PASSWORD | DISTINGUISHED NAME | CERTIFICATE [TYPE 'type'] [VERSION 'version'] } }
proxy_clause	{ GRANT | REVOKE } CONNECT THROUGH proxy [WITH { ROLE { role_name [, role_name]... | ALL EXCEPT role_name [, role_name]... } | NO ROLES }] [proxy_authentication]

Subclause	Syntax
qualified_disk_clause	search_string [NAME disk_name] [SIZE size_clause] [FORCE | NOFORCE]
qualified_template_clause	template_name ATTRIBUTES ([MIRROR | UNPROTECTED] [FINE | COARSE])
query_partition_clause	PARTITION BY { value_expr[, value_expr]... | (value_expr[, value_expr]...)
query_table_expression	{ query_name | [schema.] { table [{ PARTITION (partition) | SUBPARTITION (subpartition) } [sample_clause] | [sample_clause] | @ dblink] | { view | materialized view } [@ dblink] } | (subquery [subquery_restriction_clause]) | table_collection_expression }
quiesce_clauses	QUIESCE RESTRICTED | UNQUIESCE
range_partitioning	PARTITION BY RANGE (column[, column]...) (PARTITION [partition] range_values_clause table_partition_description [, PARTITION [partition] range_values_clause table_partition_description]...)
range_values_clause	VALUES LESS THAN ({ value | MAXVALUE } [, { value | MAXVALUE }]...)
rebalance_diskgroup_clause	REBALANCE [POWER integer]

Subclause	Syntax

rebuild_clause	REBUILD
	  [{ PARTITION partition
	  | SUBPARTITION subpartition
	  }
	  | { REVERSE | NOREVERSE }
	  ]
	  [parallel_clause
	  | TABLESPACE tablespace
	  | PARAMETERS ('ODCI_parameters')
	  | ONLINE
	  | COMPUTE STATISTICS
	  | physical_attributes_clause
	  | key_compression
	  | logging_clause
	  ]
	  [parallel_clause
	  | TABLESPACE tablespace
	  | PARAMETERS ('ODCI_parameters')
	  | ONLINE
	  | COMPUTE STATISTICS
	  | physical_attributes_clause
	  | key_compression
	  | logging_clause
	  ]...

records_per_block_clause	{ MINIMIZE | NOMINIMIZE } RECORDS_PER_BLOCK

recover_clause	{ { DISCONNECT [FROM SESSION]
	  | { TIMEOUT integer | NOTIMEOUT }
	  }
	| { NODELAY | DEFAULT DELAY | DELAY integer }
	| NEXT integer
	| { EXPIRE integer | NO EXPIRE }
	| parallel_clause
	| USING CURRENT LOGFILE
	| UNTIL CHANGE integer
	| THROUGH { [THREAD integer] SEQUENCE integer
	  | ALL ARCHIVELOG
	  | { ALL | LAST | NEXT } SWITCHOVER
	  }
	}
	  [{ DISCONNECT [FROM SESSION]
	  | { TIMEOUT integer | NOTIMEOUT }
	  }
	  | { NODELAY | DEFAULT DELAY | DELAY integer }
	  | NEXT integer
	  | { EXPIRE integer | NO EXPIRE }
	  | parallel_clause
	  | USING CURRENT LOGFILE
	  | UNTIL CHANGE integer
	  | THROUGH { [THREAD integer] SEQUENCE integer
	  | ALL ARCHIVELOG

Subclause	Syntax
		{ ALL | LAST | NEXT } SWITCHOVER }] ...
recovery_clauses	{ general_recovery | managed_standby_recovery | BEGIN BACKUP | END BACKUP }
redo_log_file_spec	['filename' | ('filename' [, 'filename']...)] [SIZE size_clause] [REUSE]
redo_thread_clauses	{ ENABLE | DISABLE } { INSTANCE 'instance_name' | [PUBLIC] THREAD integer }
reference_model	REFERENCE reference_spreadsheet_name ON (subquery) spreadsheet_column_clauses [cell_reference_options]
references_clause	REFERENCES [schema.] { object_table | view } [(column [, column]...)] [ON DELETE { CASCADE | SET NULL }] [constraint_state]
referencing_clause	REFERENCING { OLD [AS] old | NEW [AS] new | PARENT [AS] parent } [OLD [AS] old | NEW [AS] new | PARENT [AS] parent]...
register_logfile_clause	REGISTER [OR REPLACE] [PHYSICAL | LOGICAL] LOGFILE [file_specification [, file_specification]...] FOR logminer_session_name

Subclause	Syntax
relational_properties	{ column datatype [SORT] [DEFAULT expr] [inline_constraint [inline_constraint]... | inline_ref_constraint] | { out_of_line_constraint | out_of_line_ref_constraint | supplemental_logging_props } } [, { column datatype [SORT] [DEFAULT expr] [inline_constraint [inline_constraint]... | inline_ref_constraint] | { out_of_line_constraint | out_of_line_ref_constraint | supplemental_logging_props }]...
relational_table	CREATE [GLOBAL TEMPORARY] TABLE [schema.]table [(relational_properties)] [ON COMMIT { DELETE | PRESERVE } ROWS] [physical_properties] [table_properties] ;
rename_column_clause	RENAME COLUMN old_name TO new_name
rename_index_partition	RENAME { PARTITION partition | SUBPARTITION subpartition } TO new_name
rename_partition_subpart	RENAME { PARTITION | SUBPARTITION } current_name TO new_name
replace_type_clause	REPLACE [invoker_rights_clause] AS OBJECT (attribute datatype [, attribute datatype]... [, element_spec [, element_spec]...])
resize_disk_clauses	RESIZE { ALL [SIZE size_clause] | DISK disk_name [SIZE size_clause] [, disk_name [SIZE size_clause]]... | DISKS IN FAILGROUP

Subclause	Syntax
	failgroup_name [SIZE size_clause] [, failgroup_name [SIZE size_clause]]... }
resource_parameters	{ { SESSIONS_PER_USER | CPU_PER_SESSION | CPU_PER_CALL | CONNECT_TIME | IDLE_TIME | LOGICAL_READS_PER_SESSION | LOGICAL_READS_PER_CALL | COMPOSITE_LIMIT } { integer | UNLIMITED | DEFAULT } | PRIVATE_SGA { integer [K | M] | UNLIMITED | DEFAULT } }
restricted_session_clauses	{ ENABLE | DISABLE } RESTRICTED SESSION
return_clause	{ RETURN datatype [{ IS | AS } call_spec] | sqlj_object_type_sig }
return_rows_clause	RETURN { UPDATED | ALL } ROWS
returning_clause	RETURNING expr [, expr]... INTO data_item [, data_item]...
revoke_object_privileges	{ object_privilege | ALL [PRIVILEGES] } [, { object_privilege | ALL [PRIVILEGES] }]... on_object_clause FROM grantee_clause [CASCADE CONSTRAINTS | FORCE]
revoke_system_privileges	{ system_privilege | role | ALL PRIVILEGES } [, { system_privilege | role | ALL PRIVILEGES }]... FROM grantee_clause
rollup_cube_clause	{ ROLLUP | CUBE } (grouping_expression_list)

Subclause	Syntax
routine_clause	[schema.] [type. | package.] { function | procedure | method } [@dblink_name] ([argument [, argument]...])
row_movement_clause	{ ENABLE | DISABLE } ROW MOVEMENT
sample_clause	SAMPLE [BLOCK] (sample_percent) [SEED (seed_value)]
schema_object_clause	{ object_option [, object_option]... | ALL } auditing_on_clause
scoped_table_ref_constraint	{ SCOPE FOR ({ ref_column | ref_attribute }) IS [schema.] { scope_table_name | c_alias } } [, SCOPE FOR ({ ref_column | ref_attribute }) IS [schema.] { scope_table_name | c_alias }]...
searched_case_expression	WHEN condition THEN return_expr [WHEN condition THEN return_expr]...
security_clause	GUARD { ALL | STANDBY | NONE }
segment_attributes_clause	{ physical_attributes_clause | TABLESPACE tablespace | logging_clause } [physical_attributes_clause | TABLESPACE tablespace | logging_clause]...
segment_management_clause	SEGMENT SPACE MANAGEMENT { MANUAL | AUTO }
select_list	{ * | { query_name.* | [schema.] { table | view | materialized view } .* | expr [[AS] c_alias] } [, { query_name.* | [schema.] { table | view | materialized view } .*

Subclause	Syntax
		expr [[AS] c_alias] }]... }
set_subpartition_template	SET SUBPARTITION TEMPLATE { (SUBPARTITION subpartition [list_values_clause] [partitioning_storage_clause] [, SUBPARTITION subpartition [list_values_clause] [partitioning_storage_clause]...]) | hash_subpartition_quantity }
set_time_zone_clause	SET TIME_ZONE = '{ { + | - } hh : mi | time_zone_region }'
shrink_clause	SHRINK SPACE [COMPACT] [CASCADE]
shutdown_dispatcher_clause	SHUTDOWN [IMMEDIATE] dispatcher_name
simple_case_expression	expr WHEN comparison_expr THEN return_expr [WHEN comparison_expr THEN return_expr]...
single_column_for_loop	FOR dimension_column { IN ({ literal [, literal]... | subquery }) | [LIKE pattern] FROM literal TO literal { INCREMENT | DECREMENT } literal }
single_table_insert	insert_into_clause { values_clause [returning_clause] | subquery }
size_clause	integer [K | M | G | T]

Subclause	Syntax
split_index_partition	SPLIT PARTITION partition_name_old AT (value [, value]...) [INTO (index_partition_description, index_partition_description)] [parallel_clause]
split_table_partition	SPLIT PARTITION current_partition { AT | VALUES } (value [, value]...) [INTO (partition_spec, partition_spec)] [update_index_clauses] [parallel_clause]
split_table_subpartition	SPLIT SUBPARTITION subpartition VALUES ({ value | NULL } [, value | NULL]...) [INTO (subpartition_spec, subpartition_spec)] [update_index_clauses] [parallel_clause]
sql_statement_clause	{ { statement_option | ALL } [, { statement_option | ALL }]... | { system_privilege | ALL PRIVILEGES } [, { system_privilege | ALL PRIVILEGES }]... } [auditing_by_clause]
sqlj_object_type	EXTERNAL NAME java_ext_name LANGUAGE JAVA USING (SQLData | CustomDatum | OraData)
sqlj_object_type_attr	EXTERNAL NAME 'field_name'
sqlj_object_type_sig	RETURN { datatype | SELF AS RESULT } EXTERNAL { VARIABLE NAME 'java_static_field_name' | NAME 'java_method_sig' }
standby_database_clauses	(activate_standby_db_clause | maximize_standby_db_clause | register_logfile_clause | commit_switchover_clause | start_standby_clause | stop_standby_clause) [parallel_clause]

Subclause	Syntax
start_standby_clause	START LOGICAL STANDBY APPLY [IMMEDIATE] [NODELAY] [NEW PRIMARY dblink | INITIAL [scn_value] | { SKIP FAILED TRANSACTION | FINISH }]
startup_clauses	{ MOUNT [{ STANDBY | CLONE } DATABASE] | OPEN { [READ WRITE] [RESETLOGS | NORESETLOGS] [UPGRADE | DOWNGRADE] | READ ONLY } }
stop_standby_clause	{ STOP | ABORT } LOGICAL STANDBY APPLY
storage_clause	STORAGE ({ INITIAL integer [K | M] | NEXT integer [K | M] | MINEXTENTS integer | MAXEXTENTS { integer | UNLIMITED } | PCTINCREASE integer | FREELISTS integer | FREELIST GROUPS integer | OPTIMAL [integer [K | M] | NULL] | BUFFER_POOL { KEEP | RECYCLE | DEFAULT } } [INITIAL integer [K | M] | NEXT integer [K | M] | MINEXTENTS integer | MAXEXTENTS { integer | UNLIMITED } | PCTINCREASE integer | FREELISTS integer | FREELIST GROUPS integer | OPTIMAL [integer [K | M] | NULL] | BUFFER_POOL { KEEP | RECYCLE | DEFAULT }]...)
streaming_clause	{ ORDER | CLUSTER } BY (column [, column]...)
subpartition_by_hash	SUBPARTITION BY HASH (column [, column]...) [SUBPARTITIONS quantity [STORE IN (tablespace [, tablespace]...)]

Subclause	Syntax
		subpartition_template]
subpartition_by_list	SUBPARTITION BY LIST (column) [subpartition_template]
subpartition_spec	SUBPARTITION [subpartition] [list_values_clause] [partitioning_storage_clause]
subpartition_template	SUBPARTITION TEMPLATE (SUBPARTITION subpartition [list_values_clause] [partitioning_storage_clause] [, SUBPARTITION subpartition [list_values_clause] [partitioning_storage_clause]]) | hash_subpartition_quantity
subprogram_declaration	{ MEMBER | STATIC } { procedure_declaration | function_declaration | constructor_declaration }
subprogram_spec	{ MEMBER | STATIC } { procedure_spec | function_spec }
subquery	[subquery_factoring_clause] SELECT [hint] [{ { DISTINCT | UNIQUE } | ALL }] select_list FROM table_reference [, table_reference]... [where_clause] [hierarchical_query_clause] [group_by_clause] [HAVING condition] [model_clause] [{ UNION [ALL] | INTERSECT | MINUS } (subquery)

Subclause	Syntax
	] [order_by_clause]
subquery_factoring_clause	WITH query_name AS (subquery) [, query_name AS (subquery)]...
subquery_restriction_clause	WITH { READ ONLY | CHECK OPTION [CONSTRAINT constraint] }
substitutable_column_clause	[ELEMENT] IS OF [TYPE] ([ONLY] type) | [NOT] SUBSTITUTABLE AT ALL LEVELS
supplemental_db_logging	{ ADD | DROP } SUPPLEMENTAL LOG { DATA | supplemental_id_key_clause }
supplemental_id_key_clause	DATA ({ ALL | PRIMARY KEY | UNIQUE | FOREIGN KEY } [, { ALL | PRIMARY KEY | UNIQUE | FOREIGN KEY }]...) COLUMNS
supplemental_log_grp_clause	GROUP log_group (column [NO LOG] [, column [NO LOG]]...) [ALWAYS]
supplemental_logging_props	{ supplemental_log_grp_clause | supplemental_id_key_clause }
supplemental_table_logging	{ ADD SUPPLEMENTAL LOG { supplemental_log_grp_clause | supplemental_id_key_clause } [, SUPPLEMENTAL LOG { supplemental_log_grp_clause | supplemental_id_key_clause }

Subclause	Syntax
	]... | DROP SUPPLEMENTAL LOG { supplemental_id_key_clause | GROUP log_group } [, SUPPLEMENTAL LOG { supplemental_id_key_clause | GROUP log_group }]... }
table_collection_expression	TABLE (collection_expression) [(+)]
table_compression	{ COMPRESS | NOCOMPRESS }
table_index_clause	[schema.]table [t_alias] (index_expr [ASC | DESC] [, index_expr [ASC | DESC]]...) [index_properties]
table_partition_description	[segment_attributes_clause] [table_compression | key_compression] [OVERFLOW [segment_attributes_clause]] [{ LOB_storage_clause | varray_col_properties } [LOB_storage_clause | varray_col_properties]...] [partition_level_subpartition]
table_partitioning_clauses	{ range_partitioning | hash_partitioning | list_partitioning | composite_partitioning }
table_properties	[column_properties] [table_partitioning_clauses] [CACHE | NOCACHE] [parallel_clause] [ROWDEPENDENCIES | NOROWDEPENDENCIES] [enable_disable_clause] [enable_disable_clause]... [row_movement_clause] [AS subquery]

Subclause	Syntax

table_reference	{ ONLY
	  (query_table_expression)
	  [flashback_query_clause]
	  [t_alias]
	| query_table_expression
	  [flashback_query_clause]
	  [t_alias]
	| (join_clause)
	| join_clause
	}

tablespace_clauses	{ EXTENT MANAGEMENT LOCAL
	| DATAFILE file_specification
	  [, file_specification]...
	| SYSAUX DATAFILE file_specification
	  [, file_specification]...
	| default_tablespace
	| default_temp_tablespace
	| undo_tablespace
	}

tablespace_group_clause	TABLESPACE GROUP { tablespace_group_name | '' }
tablespace_logging_clauses	{ logging_clause | [NO] FORCE LOGGING }
tablespace_retention_clause	RETENTION { GUARANTEE | NOGUARANTEE }
tablespace_state_clauses	{ ONLINE | OFFLINE [NORMAL | TEMPORARY | IMMEDIATE] } | READ { ONLY | WRITE } | { PERMANENT | TEMPORARY }

temporary_tablespace_clause	TEMPORARY TABLESPACE tablespace
	  [TEMPFILE file_specification
	  [, file_specification]...
	  ]
	  [tablespace_group_clause]
	  [extent_management_clause]

text	[N | n] { 'c [c]...' | { Q | q } 'quote_delimiter c [c]... quote_delimiter' }

Subclause	Syntax
trace_file_clause	TRACE [AS 'filename' [REUSE]] [RESETLOGS | NORESETLOGS]
truncate_partition_subpart	TRUNCATE { PARTITION partition | SUBPARTITION subpartition } [{ DROP | REUSE } STORAGE] [update_index_clauses [parallel_clause]]
undo_tablespace	[BIGFILE | SMALLFILE] UNDO TABLESPACE tablespace [TABLESPACE file_specification [, file_specification]...]
undo_tablespace_clause	UNDO TABLESPACE tablespace [DATAFILE file_specification [, file_specification]...] [extent_management_clause] [tablespace_retention_clause]
undrop_disk_clause	UNDROP DISKS
update_all_indexes_clause	UPDATE INDEXES [(index ({ update_index_partition | update_index_subpartition })) [, (index ({ update_index_partition | update_index_subparition }))]...
update_global_index_clause	{ UPDATE | INVALIDATE } GLOBAL INDEXES
update_index_clauses	{ update_global_index_clause | update_all_indexes_clause }
update_index_partition	PARTITION [partition] [index_partition_description [index_subpartition_clause]] [, PARTITION [partition] [index_partition_description

Subclause	Syntax
	[index_subpartition_clause]]]...
update_index_subpartition	SUBPARTITION [subpartition] [TABLESPACE tablespace] [, SUBPARTITION [subpartition] [TABLESPACE tablespace]]...
update_set_clause	SET { { (column [, column]...) = (subquery) | column = { expr | (subquery) | DEFAULT } } [, { (column [, column]...) = (subquery) | column = { expr | (subquery) | DEFAULT } }]... | VALUE (t_alias) = { expr | (subquery) } }
upgrade_table_clause	UPGRADE [[NOT] INCLUDING DATA] [column_properties]
using_function_clause	USING [schema.] [package. | type.]function_name
using_index_clause	USING INDEX { [schema.]index | (create_index_statement) | index_properties }
using_statistics_type	USING { [schema.] statistics_type | NULL }
using_type_clause	USING [schema.]implementation_type [array_DML_clause]
validation_clauses	{ VALIDATE REF UPDATE [SET DANGLING TO NULL] | VALIDATE STRUCTURE [CASCADE] [into_clause] { OFFLINE| ONLINE } }

Subclause	Syntax
values_clause	VALUES ({ expr | DEFAULT } [, { expr | DEFAULT }]...)
varray_col_properties	VARRAY varray_item { [substitutable_column_clause] STORE AS LOB { [LOB_segname] (LOB_parameters) | LOB_segname } | substitutable_column_clause }
where_clause	WHERE condition
windowing_clause	{ ROWS | RANGE } { BETWEEN { UNBOUNDED PRECEDING | CURRENT ROW | value_expr { PRECEDING | FOLLOWING } } AND { UNBOUNDED FOLLOWING | CURRENT ROW | value_expr { PRECEDING | FOLLOWING } } | { UNBOUNDED PRECEDING | CURRENT ROW | value_expr PRECEDING } }
XML_attributes_clause	XMLATTRIBUTES (value_expr [AS c_alias] [, value_expr [AS c_alias]...)
XMLSchema_spec	[XMLSCHEMA XMLSchema_URL] ELEMENT { element | XMLSchema_URL # element }
XMLType_column_properties	XMLTYPE [COLUMN] column [XMLType_storage] [XMLSchema_spec]
XMLType_storage	STORE AS { OBJECT RELATIONAL | CLOB [{ LOB_segname [(LOB_parameters)] | LOB_parameters }]

Subclause	Syntax
XMLType_table	CREATE TABLE [GLOBAL TEMPORARY] TABLE [schema.]table OF XMLTYPE [(oject_properties)] [XMLTYPE XMLType_storage] [XMLSchema_spec] [ON COMMIT { DELETE | PRESERVE } ROWS] [OID_clause] [OID_index_clause] [physical_properties] [table_properties] ;
XMLType_view_clause	OF XMLTYPE [XMLSchema_spec] WITH OBJECT IDENTIFIER { DEFAULT | (expr [, expr]...) }