

Chapter 12

Create, Alter, Drop Tables

Object Naming Conventions

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

2

Creating a Table

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

- Many optional clauses for creating tables we don't cover
- Two ways to create
 - specify each column's name and data type
 - base the new table on columns in another existing table
- User must have CREATE TABLE privilege (Ch 15)
- Column names are unique within a table
- Oracle permits up to 1,000 columns per table
- DEFAULT expr
 - specifies a value to use when no field value is explicitly provided when a new row is INSERTed

3

Commonly Used Oracle Data Types

Datatype	Description
VARCHAR2(size)	Variable-length character data (≤4,000)
CHAR(size)	Fixed-length character data (≤2,000)
NUMBER(prec,scale)	Variable-length numeric data
DATE	Date & Time (1/1/4712 BC – 12/31/9999 AD)
LONG (deprecated)	Variable-length character data up to 2 gigabytes
CLOB	Single-byte 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

4

CREATE TABLE Example #1

```
CREATE TABLE writer(  
  writerid CHAR(4) ,  
  ln VARCHAR2(30) CONSTRAINT writer_ln_nn NOT NULL,  
  fn VARCHAR2(20) CONSTRAINT writer_fn_nn NOT NULL,  
  phone VARCHAR2(14) ,  
  lastcontact DATE ,  
  freelancer CHAR(1) ,  
  amount NUMBER(6,2) ,  
  contact CHAR(4) ,  
  CONSTRAINT writer_writerid_pk PRIMARY KEY (writerid) ,  
  CONSTRAINT writer_contact_fk  
    FOREIGN KEY(contact) REFERENCES writer(writerid) ,  
  CONSTRAINT writer_freelancer_ck CHECK(freelancer IN ('Y','N')));
```

5

CREATE TABLE Example #2

```
CREATE TABLE article (  
  articlenum NUMBER(4) CONSTRAINT article_articlenum_pk PRIMARY KEY,  
  title VARCHAR2(50) CONSTRAINT article_title_nn NOT NULL ,  
  type CHAR(3) ,  
  issue DATE ,  
  length NUMBER(5) ,  
  writerid CHAR(4) CONSTRAINT article_writerid_nn NOT NULL ,  
  CONSTRAINT article_writerid_fk  
    FOREIGN KEY(writerid) REFERENCES writer(writerid)  
    ON DELETE CASCADE ,  
  CONSTRAINT article_type_fk FOREIGN KEY(type) REFERENCES type(type) );
```

6

Integrity Constraints

- For enforcing **business rules**
 - help assure data integrity
- Goal: prevent incomplete, incorrect data from being stored
- Two locations to declare constraints in CREATE TABLE
 - Column-level constraint
 - cited as the column is being defined, after its data type
 - NOT NULL constraints must be declared in this manner
 - Table-level constraint
 - cited after all of the individual columns have been defined
 - all constraints except NON NULL can be declared in this manner
 - foreign keys and concatenated primary keys must be declared in this manner
 - Most constraints can be equivalently declared either location

7

Integrity Constraints

- **Primary Key Constraint**
 - Requires that the column's values be non-null and unique
 - Oracle automatically creates an **index** for the primary key
 - the index aids in quickly identifying duplicate values and disallows nulls
- **Unique Constraint**
 - No duplicate values allowed within the column
 - NULLS are allowed
 - Oracle automatically creates an **index** to enforce
 - Commonly created for _____ key columns

8

Integrity Constraints

Foreign Key Constraint

- used to enforce **referential integrity**
 - requires each non-null **FK** to match the value of a parent table row's **PK**
- must be declared as table-level constraints
- the foreign key field must have the same data type as the primary key it references
- **ON DELETE CASCADE** clause
 - indicates that when a parent row is deleted, the matching row(s) in the child table are to be deleted too
- **ON DELETE SET NULL** clause
 - indicates that when a parent row is deleted, the foreign keys in the matching row(s) in the child table are to be set to Null
- **ON DELETE RESTRICT** clause (default)
 - the row in the parent table cannot be deleted when there are matching rows in the child table

9

Integrity Constraints

Check Constraint

- Used to limit the value entered into a field
- An expression that is evaluated as either TRUE or FALSE
- Oracle rejects the row if the constraint evaluates FALSE
- Cannot refer to column in another table
- Cannot use a subquery

Not Null Constraint

- Prevents a new or modified record from being stored if the specified field is not assigned a value

10

Naming Constraints

- A Good Habit
 - makes it easier to understand constraint violation error messages
- General Form
 - tablename_fieldname_type
- Examples
 - article_articlenum_pk
 - article_writerid_fk
 - writer_freelancer_ck

```
CREATE TABLE x (f1 NUMBER(9) PRIMARY KEY);

SELECT constraint_name
FROM USER_CONSTRAINTS
WHERE table_name = 'X';
```

Statement Output x Query Result x

SQL | All Rows Fetched: 1 in 0 seconds

CONSTRAINT_NAME
1 SYS_C0055257

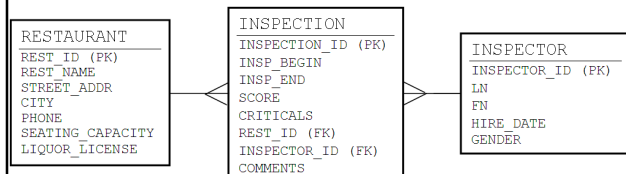
- When you don't name a constraint, Oracle will name it for you

11

Practice Time: Restaurant Inspector Design

Business Rules

- Each restaurant is usually inspected at least twice a year.
- Each inspection is made by one inspector.
- Inspectors report an overall score (0 to 100, with 70 or higher to pass), the number of critical violations, and can include detailed comments of up to 2,500 characters.
- An inspection must begin before it can end.



12

Creating a Table From an Existing Table

- Columns in the new table inherit the same definition as the fields selected from the existing table
- The new table can contain rows from the existing table(s)

```
CREATE TABLE freelancer AS
SELECT ln, fn, phone
FROM writer
WHERE freelancer = 'Y';
```

```
CREATE TABLE freelancer2 AS
SELECT ln, fn, phone
FROM writer
WHERE 'a' = 'g';
```

- Most constraints are not inherited
 - only NOT NULL constraints are inherited
 - must define primary key, foreign key, check, unique constraints

13

Renaming a Table

```
RENAME current_name TO new_name
```

```
RENAME freelancer TO nonstaff;
```

- Watch out for dependency problems
 - Constraints are retained and are automatically adjusted
 - if the table you rename is referenced by foreign keys of another table, the foreign key constraint definitions are modified to refer to the table by its new name
 - the constraint names are not adjusted
 - Views and program units (stored procedures, stored functions, packages) that use the table's old name become invalid (not usable)

14

Dropping a Table

```
DROP TABLE table [CASCADE CONSTRAINTS] [PURGE]
```

```
DROP TABLE nonstaff;
DROP TABLE x;
DROP TABLE freelancer2 PURGE;
```

- The table's structure and data rows are moved into the **Recycle Bin** (unless PURGE clause is used)
- All of the table's permissions, constraints, indexes, and triggers are also dropped
- You cannot roll back this statement!
- Any pending transaction is committed
 - each DDL statement generates an implicit COMMIT
- How is **DROP TABLE** different from **TRUNCATE**?

15

Dropping a Table

```
DROP TABLE table [CASCADE CONSTRAINTS] [PURGE]
```

- Watch out for dependency problems
 - If the table you are dropping is referenced by **foreign keys** of another table, you must include the **CASCADE CONSTRAINTS** option
 - Oracle will drop any foreign key constraints that referred to a column in the dropped table
 - Views and PL/SQL program units that depend on a dropped table remain, but become invalid (not usable)

```
DROP TABLE writer;
```

Statement Output x

Error starting at line 63 in command:
DROP TABLE writer;
Error report:
SQL Error: ORA-02449: unique/primary keys in table referenced by foreign keys
02449. 00000 - "unique/primary keys in table referenced by foreign keys"

16

Recycle Bin (new 10g)



- Dropped tables are moved into the Recycle Bin
- **USER_RECYCLEBIN** View

- see objects contained in the Recycle Bin

```
SELECT object_name, original_name, type, droptime FROM USER_RECYCLEBIN;
```

	OBJECT_NAME	ORIGINAL_NAME	TYPE	DROPTIME
1	BIN\$ouWahqiIT7u+89QxwH051A==#0	NONSTAFF	TABLE	2011-04-11:16:11:59
2	BIN\$nzag13NPRkqsJocXISmUHYw==#0	SYS_C0055257	INDEX	2011-04-11:16:12:00
3	BIN\$UuoztazERx2U1101TzB89w==#0	X	TABLE	2011-04-11:16:12:00

- Purge the Recycle Bin to remove dropped objects and free the disk space they occupied

```
--PURGE RECYCLEBIN;
```

17

Flashback Table (new 10g)

```
FLASHBACK TABLE table [, tablename...]  
TO {TIMESTAMP expr | BEFORE DROP [RENAME TO newtablename]}
```

- Restores a table's data to an earlier state
 - also retrieves the table's indexes, triggers, and constraints except for referential integrity constraints that reference other tables
- Oracle cannot restore a table to a state earlier than the most recent change to the table's structure
- Oracle cannot restore a table to a state earlier than the undo data available
- Must have sufficient privileges (Ch15)
- Examples

```
FLASHBACK TABLE nonstaff TO BEFORE DROP;
```

- TO TIMESTAMP is not available in Oracle Express Edition

```
FLASHBACK TABLE writer  
TO TIMESTAMP (SYSTIMESTAMP-INTERVAL '5' minute);
```

18

Flashback Table (new 10g)

```
FLASHBACK TABLE table [, tablename...]  
TO {TIMESTAMP expr | BEFORE DROP [RENAME TO newtablename]}
```

- **Practice Time**
 - Confirm that nonstaff is in the Recycle Bin
 - Restore the nonstaff from the Recycle Bin
 - Display its rows
 - What is now in the Recycle Bin?

19

Documenting Tables and Columns (pg 534)

- Use COMMENT to add a comment about a table, view, or column into the **Data Dictionary** (ch 14)
- Excerpts from create_student script

```
COMMENT ON TABLE course IS 'Information for a course.';  
COMMENT ON COLUMN course.course_no  
IS 'The unique ID for a course.';  
COMMENT ON COLUMN course.cost  
IS 'The dollar amount charged for enrollment.';
```

- View comments by using data dictionary views

```
SELECT * FROM USER_TAB_COMMENTS WHERE table_name = 'COURSE';  
SELECT * FROM USER_COL_COMMENTS WHERE table_name = 'COURSE';
```

- To drop a comment, set it to an empty string ''

```
COMMENT ON COLUMN course.cost IS '';
```

20

Adding Columns

```
ALTER TABLE table
ADD (column datatype [DEFAULT expr]
[, column datatype]...);
```

- Can only add a new NOT NULL column when either:
 - the column has a DEFAULT value OR
 - the table has no existing rows

```
ALTER TABLE writer
ADD (sex CHAR(1) DEFAULT 'M'
CONSTRAINT writer_sex_nn NOT NULL
CONSTRAINT writer_sex_ck CHECK (sex IN ('M', 'F')));
```

ALTER TABLE writer succeeded.

- Practice Time**
 - add new ssn field to writer as a candidate key

21

Renaming a Column

```
ALTER TABLE table
RENAME column TO column
```

- Used to change a column's name
- The column's data remains intact
- Watch out for dependency problems
 - eg: views that use the column will no longer work

```
ALTER TABLE writer
RENAME COLUMN sex to gender;
```

desc writer

Name	Null	Type
WRITERID	NOT NULL	CHAR(4)
LN	NOT NULL	VARCHAR2(4)
FN	NOT NULL	VARCHAR2(4)
PHONE		VARCHAR2(15)
LASTCONTACT		DATE
FREELANCER		CHAR(1)
AMOUNT		NUMBER(6,2)
CONTACT		CHAR(4)
GENDER	NOT NULL	CHAR(1)
SSN		CHAR(9)

Dropping Columns

```
ALTER TABLE table
DROP (column, [column]...)
```

- Removes the column(s) and removes its data from each row of the table, freeing space in the data blocks on disk
- Can't drop the primary key column(s)
- At least one column must remain

```
ALTER TABLE writer DROP COLUMN gender;
```

desc writer

Name	Null	Type
WRITERID	NOT NULL	CHAR(4)
LN	NOT NULL	VARCHAR2(4)
FN	NOT NULL	VARCHAR2(4)
PHONE		VARCHAR2(15)
LASTCONTACT		DATE
FREELANCER		CHAR(1)
AMOUNT		NUMBER(6,2)
CONTACT		CHAR(4)
SSN		CHAR(9)

Dropping Columns

- Dropping a column can take considerable time
- can require lots of disk I/O
- a quicker alternative is to mark a column as unused, then later use DROP UNUSED COLUMNS to remove those columns that have been marked UNUSED

```
ALTER TABLE writer SET UNUSED (gender)
```

```
ALTER TABLE writer DROP UNUSED COLUMNS
```

24

Modifying a Column

```
ALTER TABLE table
MODIFY (column datatype [DEFAULT expr]
[, column datatype]...);
```

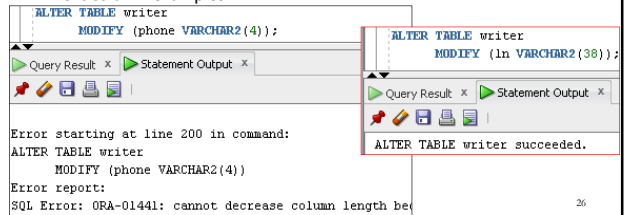
- Used to change a column's:
 - width
 - datatype
 - default value

```
ALTER TABLE writer
MODIFY (amount NUMBER(7,2)
CONSTRAINT writer_amount_nn NOT NULL,
ln VARCHAR2(40))
```

25

Changing a Text Column's Size

- You can **increase** the size of a text column
- You can **reduce** the size of a text column provided it does not cause data loss
 - Oracle scans the existing data values in the column and returns an error if data exists that exceeds the new limit
 - <ver 9i, columns containing data couldn't be decreased (ORA-01441 error)
- Text column examples



```
ALTER TABLE writer
MODIFY (phone VARCHAR2(4));
```

```
ALTER TABLE writer
MODIFY (ln VARCHAR2(38));
```

Error starting at line 200 in command:
ALTER TABLE writer
MODIFY (phone VARCHAR2(4))
Error report:
SQL Error: ORA-01441: cannot decrease column length be

ALTER TABLE writer
MODIFY (ln VARCHAR2(38));
ALTER TABLE writer succeeded.

26

Changing a Numeric Column's Size

- Can **increase** the precision or scale of a numeric column
- Can only **reduce** the precision or scale of a numeric column when all rows contain Null
 - Oracle won't let you decrease a numeric column's width even if doing so would not result in data loss
- Numeric column examples

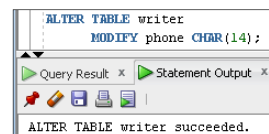
```
DESC writer
ALTER TABLE writer
MODIFY (amount NUMBER(9,2));
Table altered.
ALTER TABLE writer
MODIFY (amount NUMBER(8,2));
ORA-01440: column to be modified must be empty to
decrease precision or scale
```

- Can you think of a **workaround** to decrease a numeric size to save space when there is data is present in the column?

27

Changing a Column's Datatype

- Permitted under two circumstances:
 - the column is empty
 - changing to a **compatible** datatype
 - eg: VARCHAR2 to CHAR



```
ALTER TABLE writer
MODIFY phone CHAR(14);
```

ALTER TABLE writer succeeded.

28

Changing Column's Default Value

- Has no effect on existing rows
 - affects subsequent INSERTs into the table
- Can also eliminate a NOT NULL restriction
- Example

```
ALTER TABLE writer
  MODIFY (gender DEFAULT 'F',
         ln NULL);
```

29

Adding/Dropping/Renaming Constraints

- Add new constraint
 - a constraint is automatically enabled when it is created
 - can't add a constraint when the table's existing data violates the constraint
 - Examples:

```
ALTER TABLE writer
  ADD CONSTRAINT writer_writetid_pk PRIMARY KEY(writetid);
ALTER TABLE writer
  ADD CONSTRAINT writer_gender_ck CHECK(gender IN ('M','F'));
```

- Drop a constraint

```
ALTER TABLE writer
  DROP CONSTRAINT writer_writetid_pk;
```

- Rename a Constraint

- the constraint's definition is unchanged

```
ALTER TABLE writer
  RENAME CONSTRAINT writer_writetid_ok TO writer_writetid_pk;
```

Composite PK and FK Constraints (pg 550)

- Composite keys can be declared in a CREATE TABLE statement by placing them in table-level constraints at the bottom of the statement
- Composite key constraints can also be created after the table's creation, preferably before any rows are INSERTed

```
ALTER TABLE enrollment
  ADD CONSTRAINT enr_pk
  PRIMARY KEY (student_id, section_id);
```

```
ALTER TABLE grade
  ADD CONSTRAINT gr_enr_fk
  FOREIGN KEY (student_id, section_id)
  REFERENCES enrollment (student_id, section_id);
```

Column Name	Data Type	Nullability	Other Attributes
STUDENT_ID	NUMBER(8,0)	NOT NULL	(PK)(FK)
SECTION_ID	NUMBER(8,0)	NOT NULL	(FK)
GRADE_TYPE_CODE	CHAR(2)	NOT NULL	(FK)
GRADE_CODE_OCCURRENCE	NUMBER(38,0)	NOT NULL	(FK)
NUMERIC_GRADE	NUMBER(3,0)	NOT NULL	
COMMENTS	VARCHAR2(2000)	NULL	
CREATED_BY	VARCHAR2(30)	NOT NULL	
CREATED_DATE	DATE	NOT NULL	
MODIFIED_BY	VARCHAR2(30)	NOT NULL	
MODIFIED_DATE	DATE	NOT NULL	

Column Name	Data Type	Nullability	Other Attributes
STUDENT_ID	NUMBER(8,0)	NOT NULL	(FK)
SECTION_ID	NUMBER(8,0)	NOT NULL	(FK)
ENROLL_DATE	DATE	NOT NULL	
INITIAL_GRADE	NUMBER(3,0)	NULL	
CREATED_BY	VARCHAR2(30)	NOT NULL	
CREATED_DATE	DATE	NOT NULL	
MODIFIED_BY	VARCHAR2(30)	NOT NULL	
MODIFIED_DATE	DATE	NOT NULL	

GR_ENR_FK

31

Enabling/Disabling Constraints

- Generally dangerous to disable constraints
- Example: importing rows
 - production table
 - staging table (w/identical field structure and constraints)

- truncate staging table
- disable constraints in staging table
- import raw rows into staging table
- data cleansing/transformation to staging table's rows
- enable constraints in staging table

```
ALTER TABLE writer_staging
  DISABLE CONSTRAINT writer_staging_gender_ck;
```

```
ALTER TABLE writer_staging
  ENABLE CONSTRAINT writer_staging_gender_ck;
```

- append rows from staging table to production table

32