Database Programming with SQL

13-1: Creating Tables

1. Complete the GRADUATE CANDIDATE table instance chart. Credits is a foreign-key column referencing the requirements table.

Column Name	Student_id	Last_name	First_name	Credits	Graduation_date
Key Type	Primary Key	NUMBER	6	Not Null	
Nulls/Unique					
FK Column					
Datatype	NUMBER	VARCHAR2	VARCHAR2	NUMBER	DATE
Length	6			3	

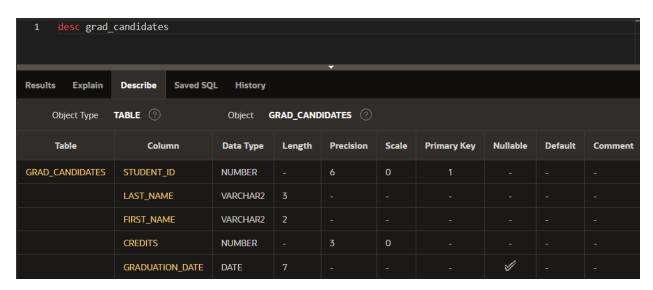
Column Name	Key Type	Datatype	Length	Nulls/Unique	FK Column
student id Primary		NUMBER	6	Not Null	
_	Key				
last_name		VARCHAR2	3	Not Null	
first_name		VARCHAR2	2	Not Null	
credits	Foreign	NUMBER	3	Not Null	references
	Key				requirements
graduation_date		DATE	7	Nullable	

2. Write the syntax to create the grad candidates table.

CREATE TABLE grad_candidates (
student_id NUMBER(6) PRIMARY KEY,
last_name VARCHAR2(3) NOT NULL,
first_name VARCHAR2(2) NOT NULL,
credits NUMBER(3) NOT NULL,
graduation_date DATE,

FOREIGN KEY (credits) REFERENCES requirements(credit_column));

3. Confirm creation of the table using DESCRIBE.



4. Create a new table using a subquery. Name the new table your last name – e.g., smith_table. Using subquery, copy grad_candidates into smith_table.



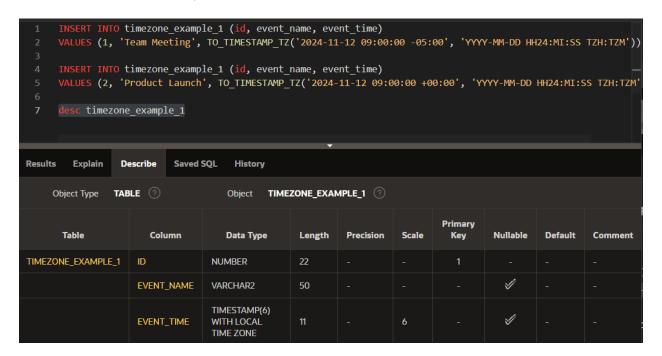
5. Insert your personal data into the table created in question 4.



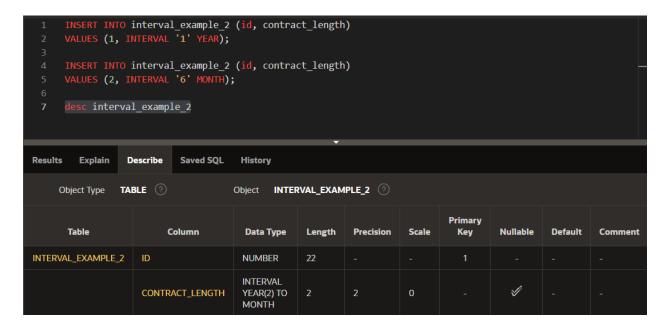
- 6. Query the data dictionary for each of the following:
 - USER TABLES: Returns tables in your schema.
 - USER OBJECTS: Returns all objects (tables, views, etc.)
 - USER_CATALOG or USER_CAT: Returns metadata about all objects in the database catalog that are accessible to the user.

13-2: Using Data Types

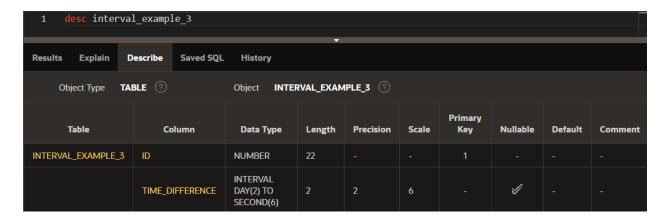
- 1. Create tables using each of the listed time-zone data types, use your time-zone and one other in your examples. Answers will vary.
 - a) TIMESTAMP WITH LOCAL TIME ZONE



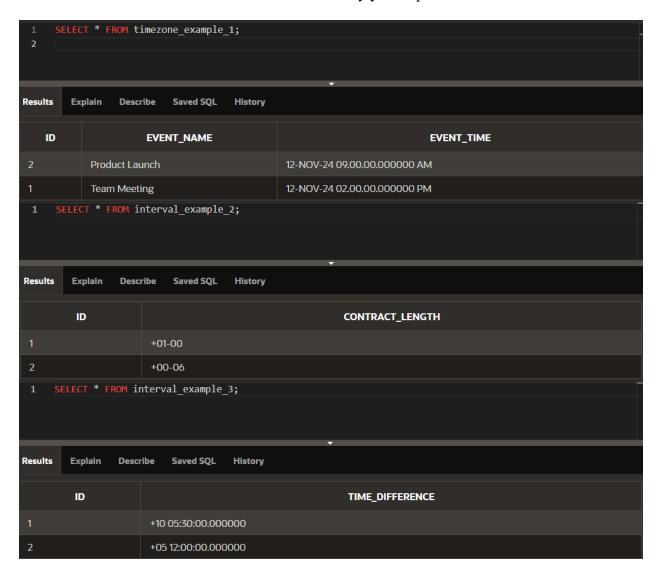
b) INTERVAL YEAR TO MONTH



c) INTERVAL DAY TO SECOND



2. Execute a SELECT * from each table to verify your input.



- 3. Give 3 examples of organizations and personal situations where it is important to know to which time zone a date-time value refers.
 - Global Businesses with Offices in Multiple Locations
 - Travel and Airline Industry
 - Financial and Stock Markets

13-3: Modifying a Table

1. Create the three o_tables – jobs, employees, and departments – using the syntax: CREATE TABLE o_jobs AS (SELECT * FROM jobs); CREATE TABLE o_employees AS (SELECT * FROM employees); CREATE TABLE o_departments AS (SELECT * FROM departments);

```
1 CREATE TABLE o_jobs AS (SELECT * FROM jobs);
2 CREATE TABLE o_employees AS (SELECT * FROM employees);
3 CREATE TABLE o_departments AS (SELECT * FROM departments);
```

2. Add the Human Resources job to the jobs table: INSERT INTO o_jobs (job_id, job_title, min_salary, max_salary) VALUES('HR_MAN', 'Human Resources Manager', 4500, 5500);

```
1 INSERT INTO o_jobs (job_id, job_title, min_salary, max_salary)
2 VALUES('HR_MAN', 'Human Resources Manager', 4500, 5500);
```

3. Add the three new employees to the employees table: INSERT INTO o_employees (employee_id, first_name, last_name, email, hire_date, job_id) VALUES(210, 'Ramon', 'Sanchez', 'RSANCHEZ', SYSDATE, 'HR_MAN');

```
INSERT INTO o_employees (employee_id, first_name, last_name, email, hire_date, job_id)
VALUES (210, 'Ramon', 'Sanchez', 'RSANCHEZ', SYSDATE, 'HR_MAN');

INSERT INTO o_employees (employee_id, first_name, last_name, email, hire_date, job_id)
VALUES (211, 'Maria', 'Garcia', 'MGARCIA', SYSDATE, 'HR_MAN');

INSERT INTO o_employees (employee_id, first_name, last_name, email, hire_date, job_id)
VALUES (212, 'Carlos', 'Lopez', 'CLopez', SYSDATE, 'HR_MAN');
```

4. Add Human Resources to the departments table: INSERT INTO o_department_id, department_name) VALUES (210,'Human Resources');

```
1 INSERT INTO o_departments(department_id, department_name)
2 VALUES (210, 'Human Resources');
```

You will need to know which columns do not allow null values.

- 1. Why is it important to be able to modify a table?
 - Data Integrity
 - Business Needs
 - Error Correction
 - Flexibility
- 2. CREATE a table called Artists.
 - a. Add the following to the table:
 - Artist ID
 - First Name
 - Last Name
 - Band Name
 - Email
 - Hourly Rate

b. INSERT one artist from the d songs table.

```
1   INSERT INTO Artists (artist_id, first_name, last_name, band_name, email)
2   SELECT ID, ARTIST, NULL, TITLE, NULL
3   FROM d_songs
4   WHERE rownum = 1;
```

c. INSERT one artist of your own choosing.

```
1 INSERT INTO Artists (artist_id, first_name, last_name, band_name, email, hourly_rate)
2 VALUES (1001, 'John', 'Doe', 'The Rockers', 'john.doe@example.com', 50);
```

- d. Give an example how each of the following may be used on the table that you have created:
 - 1) ALTER TABLE: Used to modify the structure of an existing table.
 - 2) DROP TABLE: Used to remove a table completely from the database.
 - 3) RENAME TABLE: Used to rename an existing table.
 - 4) TRUNCATE: Used to remove all rows from a table.
 - 5) COMMENT ON TABLE: Adds a comment to the table, which can be seen in the data dictionary.

3. In your o_employees table, enter a new column called "Termination." The datatype for the new column should be VARCHAR2. Set the DEFAULT for this column as SYSDATE to appear as character data in the format: February 20th, 2003.

```
1 ALTER TABLE o_employees
2 ADD (termination VARCHAR2(100) DEFAULT TO_CHAR(SYSDATE, 'Month DD, YYYY'));
```

4. Create a new column in the o_employees table called start_date. Use the TIMESTAMP WITH LOCAL TIME ZONE as the datatype.

```
1 ALTER TABLE o_employees
2 ADD (start_date TIMESTAMP WITH LOCAL TIME ZONE);
```

5. Truncate the o_jobs table. Then do a SELECT * statement. Are the columns still there? Is the data still there?

```
1 TRUNCATE TABLE o_jobs;
2
3 SELECT * FROM o_jobs;

Results Explain Describe Saved SQL History
no data found
```

- 6. What is the distinction between TRUNCATE, DELETE, and DROP for tables?
 - TRUNCATE removes all rows from the table but keeps the table structure while DELETE can be used for the same purpose except it helps you filter and DROP completely removes the table from database.
- 7. List the changes that can and cannot be made to a column.
 - Can be changed: Data Type, Column Name, Constraints
 - Cannot be Changed: Length of VARCHAR columns and some other structural changes depend on the DBMS version.
- 8. Add the following comment to the o jobs table:

"New job description added"

View the data dictionary to view your comments.



9. Rename the o jobs table to o job description.

```
1 RENAME o_jobs TO o_job_description;
```

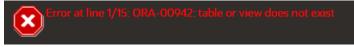
- 10. F staffs table exercises:
- a. Create a copy of the f_staffs table called copy_f_staffs and use this copy table for the remaining labs in this lesson.

b. Describe the new table to make sure it exists.

COPY_F_STAFFS	ID	NUMBER	-	5	0	1	-	-	-
	FIRST_NAME	VARCHAR2	25				 ✓		-
	LAST_NAME	VARCHAR2	35				 ✓		-
	BIRTHDATE	DATE	7				s/		-
	SALARY	NUMBER		8	2		s/		-
	OVERTIME_RATE	NUMBER		5	2		s/		-
	TRAINING	VARCHAR2	50				 ✓		-
	STAFF_TYPE	VARCHAR2	20	-	-	-		-	-

c. Drop the table.

d. Try to select from the table.



e. Investigate your recyclebin to see where the table went.

```
OBJECT_NAME

BIN$JrixUEYMBETgYxJ+eGTYgg==$0

BIN$JrqcYGcQ7UrgYxJ+eGTL7A==$0

BIN$JrroJRdeHFDgYxJ+eGT6jg==$0
```

f. Undrop the table.

```
9 FLASHBACK TABLE copy_f_staffs TO BEFORE DROP;
```

g. Describe the table.

	Object Type TABLE ? Object			COPY_F_STAFFS ?						
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment	
COPY_F_STAFFS	ID	NUMBER		5	0	1			-	
	FIRST_NAME	VARCHAR2	25				S/		-	
	LAST_NAME	VARCHAR2	35				s/		-	
	BIRTHDATE	DATE	7				S/		-	
	SALARY	NUMBER		8	2		S		-	
	OVERTIME_RATE	NUMBER		5	2		s/		-	

- 11. Still working with the copy f staffs table, perform an update on the table.
- a. Issue a select statement to see all rows and all columns from the copy_f_staffs table:



- b. Change the salary for Sue Doe to 12 and commit the change.
- c. Issue a select statement to see all rows and all columns from the copy_f_staffs table;



d. For Sue Doe, update the salary to 2 and commit the change.

e. Issue a select statement to see all rows and all columns from the copy_f_staffs table;

1	1 SELECT * FROM copy_f_staffs;										
Result	s Explain D	escribe Saved S	QL History								
ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING	STAFF_TYPE	MANAGER_ID			
12	Sue	Doe	01-Jul-1980	2	11.1		Order Taker	19			
9	Bob	Miller	19-Mar- 1979	10	.75		Grill Cook	19			
19	Monique	Tuttle	30-Mar- 1969	60			Manager	-			

f. Now, issue a FLASHBACK QUERY statement against the copy_f_staffs table, so you can see all the changes made. g. Investigate the result of f), and find the original salary and update the copy_f_staffs table salary column for Sue Doe back to her original salary.

```
1 SELECT salary FROM copy_f_staffs AS OF TIMESTAMP (SYSTIMESTAMP - INTERVAL '5' MINUTE)
2 WHERE first_name = 'Sue' AND last_name = 'Doe';
3
4 UPDATE copy_f_staffs
5 SET salary = 10
6 WHERE first_name = 'Sue' AND last_name = 'Doe';

Results Explain Describe Saved SQL History

1 row(s) updated.
```