14-1: Intro to Constraints; NOT NULL and UNIQUE Constraints

1. What is a “constraint” as it relates to data integrity?

Constraints enforce rules that maintain the consistency and reliability of data. A constraint can prevent duplicate values and ensure that a column cannot contain NULL values, or make sure that values in one table correspond to values in another table.

2. What are the limitations of constraints that may be applied at the column level and at the table level?

Column-level constraints: These constraints apply to a specific column in the table. They define rules such as NOT NULL, UNIQUE, or CHECK. The limitations include that the constraint can only be applied to a single column at a time.
Table-level constraints: These constraints are defined at the table level and can involve multiple columns, such as primary keys, foreign keys, and composite unique constraints. They are more flexible because they can span multiple columns but need to be defined after all the columns are listed in the CREATE TABLE statement.

3. Why is it important to give meaningful names to constraints?

Giving meaningful names to constraints helps with maintenance and error. When errors occur, constraints make it easier to understand which rule was violated.

4. Based on the information provided by the owners, choose a datatype for each column. Indicate the length, precision, and scale for each NUMBER datatype.

Column Name	Data Type	Length/Precision/Scale
id	NUMBER	6
name	VARCHAR2	100
date_opened	DATE	
address	VARCHAR2	200
city	VARCHAR2	100
zip/postal_code	VARCHAR2	10
phone	VARCHAR2	15
email	VARCHAR2	100
manager_id	NUMBER	6
emergency_contact	VARCHAR2	100

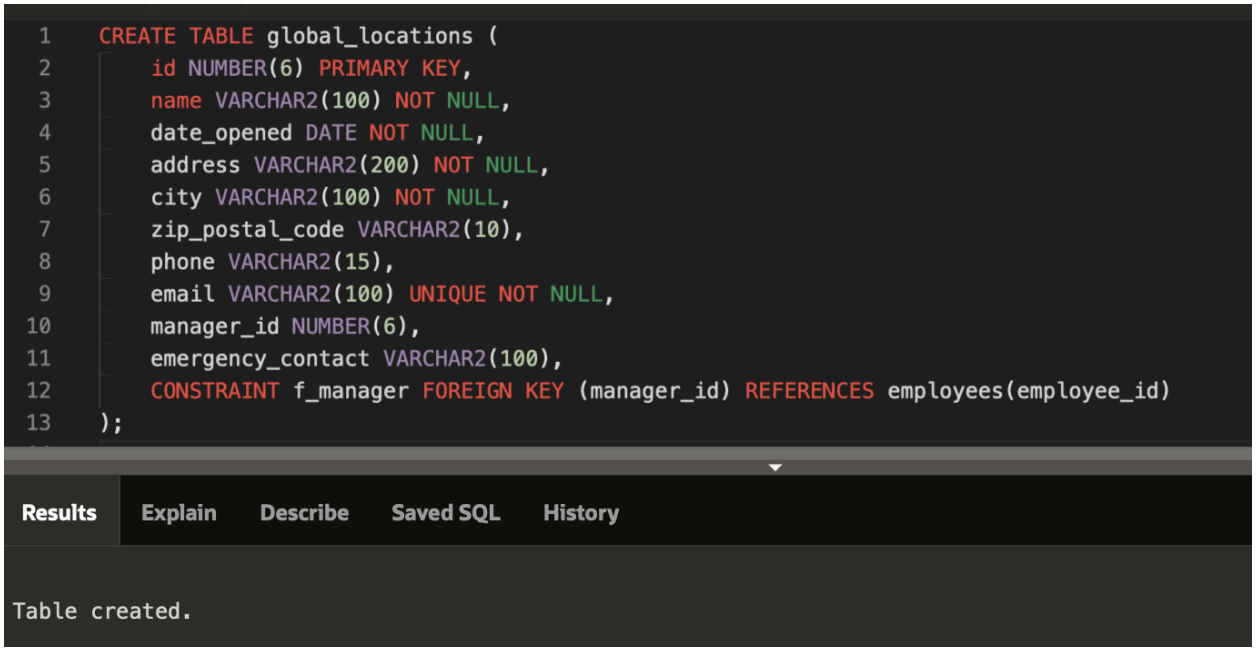
5. Use “nullable” to indicate those columns that can have null values.
   Nullable Columns: emergency_contact and manager_id (if no manager is assigned)

Non-Nullable Columns: id (Primary key, must be unique and not null), name, date_opened, address, city, zip/postal_code, phone, and email (must be unique)

6. Write the CREATE TABLE statement for the Global Fast Foods locations table to define the constraints at the column level.

```
CREATE TABLE global_locations (
   id NUMBER(6) PRIMARY KEY,
   name VARCHAR2(100) NOT NULL,
   date_opened DATE NOT NULL,
   address VARCHAR2(200) NOT NULL,
   city VARCHAR2(100) NOT NULL,
   zip_postal_code VARCHAR2(10),
   phone VARCHAR2(15),
   email VARCHAR2(100) UNIQUE NOT NULL,
   manager_id NUMBER(6),
   emergency_contact VARCHAR2(100),
   CONSTRAINT f_manager FOREIGN KEY (manager_id) REFERENCES
employees(employee_id)
);
```

7. Execute the CREATE TABLE statement in Oracle Application Express.



8. Execute a DESCRIBE command to view the Table Summary information.

```
1  DESCRIBE global_locations;
2
```

Results | Explain | Describe | Saved SQL | History

Object Type **TABLE** Object **GLOBAL_LOCATIONS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GLOBAL_LOCATIONS	ID	NUMBER	-	6	0	1	-	-	-
	NAME	VARCHAR2	100	-	-	-	-	-	-
	DATE_OPENED	DATE	7	-	-	-	-	-	-
	ADDRESS	VARCHAR2	200	-	-	-	-	-	-

9. Rewrite the CREATE TABLE statement for the Global Fast Foods locations table to define the UNIQUE constraints at the table level. Do not execute this statement.

```
CREATE TABLE global_locations (
   id NUMBER(6) PRIMARY KEY,
   name VARCHAR2(100) NOT NULL,
   date_opened DATE NOT NULL,
   address VARCHAR2(200) NOT NULL,
   city VARCHAR2(100) NOT NULL,
   zip_postal_code VARCHAR2(10),
   phone VARCHAR2(15),
   email VARCHAR2(100) NOT NULL,
   manager_id NUMBER(6),
   emergency_contact VARCHAR2(100),
   CONSTRAINT email_unique UNIQUE (email),
   CONSTRAINT f_manager FOREIGN KEY (manager_id) REFERENCES
employees(employee_id)
);
```

14-2: PRIMARY KEY, FOREIGN KEY, and CHECK Constraints

1. What is the purpose of a:

   a. PRIMARY KEY: Uniquely identifies each row in a table.

   b. FOREIGN KEY: Establishes a relationship between two tables by ensuring that the value in the foreign key column matches a value in the table's primary key or unique key.

   c. CHECK CONSTRAINT: Used to limit the range of values that can be stored in a column to ensure data validity.

2. Using the column information for the animals table below, name constraints where applicable at the table level, otherwise name them at the column level. Define the primary key (animal_id). The license_tag_number must be unique. The admit_date and vaccination_date columns cannot contain null values.

Column level constraints:

```
animal_id NUMBER(6) PRIMARY KEY,
name VARCHAR2(25),
license_tag_number NUMBER(10) UNIQUE,
admit_date DATE NOT NULL,
adoption_id NUMBER(5),
vaccination_date DATE NOT NULL
```

Table level constraints:

```
CREATE TABLE animals (
   animal_id NUMBER(6) PRIMARY KEY,
   name VARCHAR2(25),
   license_tag_number NUMBER(10) UNIQUE,
   admit_date DATE NOT NULL,
   adoption_id NUMBER(5),
   vaccination_date DATE NOT NULL
);
```

3. Create the animals table. Write the syntax you will use to create the table.

```sql
CREATE TABLE animals (
    animal_id NUMBER(6) PRIMARY KEY,
    name VARCHAR2(25),
    license_tag_number NUMBER(10) UNIQUE,
    admit_date DATE NOT NULL,
    adoption_id NUMBER(5),
    vaccination_date DATE NOT NULL
);
```

Results | Explain | Describe | Saved SQL | History

Table created.

4. Enter one row into the table. Execute a SELECT * statement to verify your input. Refer to the graphic below for input.

```sql
INSERT INTO animals (animal_id, name, license_tag_number, admit_date, adoption_id, vaccination_date)
VALUES (101, 'Spot', 35540, TO_DATE('2004-10-10', 'YYYY-MM-DD'), 205, TO_DATE('2004-12-12', 'YYYY-MM-DD'));


SELECT * FROM animals WHERE animal_id = 101;
```

Results | Explain | Describe | Saved SQL | History

ANIMAL_ID	NAME	LICENSE_TAG_NUMBER	ADMIT_DATE	ADOPTION_ID	VACCINATION_DATE
101	Spot	35540	10-Oct-2004	205	12-Dec-2004

1 rows returned in 0.00 seconds Download

5. Write the syntax to create a foreign key (adoption_id) in the animals table that has a corresponding primary- key reference in the adoptions table. Show both the column-level and

table-level syntax. Note that because you have not actually created an adoptions table, no adoption_id primary key exists, so the foreign key cannot be added to the animals table.

Column level:

```
CREATE TABLE animals (
   animal_id NUMBER(6) PRIMARY KEY,
   name VARCHAR2(25),
   license_tag_number NUMBER(10) UNIQUE,
   admit_date DATE NOT NULL,
   adoption_id NUMBER(5),
   vaccination_date DATE NOT NULL,
   CONSTRAINT fk_adoption FOREIGN KEY (adoption_id) REFERENCES
adoptions(adoption_id)
);
```

Table level:

```
CREATE TABLE animals (
   animal_id NUMBER(6) PRIMARY KEY,
   name VARCHAR2(25),
   license_tag_number NUMBER(10) UNIQUE,
   admit_date DATE NOT NULL,
   adoption_id NUMBER(5),
   vaccination_date DATE NOT NULL,
   CONSTRAINT fk_adoption FOREIGN KEY (adoption_id) REFERENCES
adoptions(adoption_id)
);
```

6. What is the effect of setting the foreign key in the ANIMAL table as:

a. ON DELETE CASCADE: When a row in the parent table (adoptions) is deleted, all corresponding rows in the child table (animals) will also be deleted automatically.

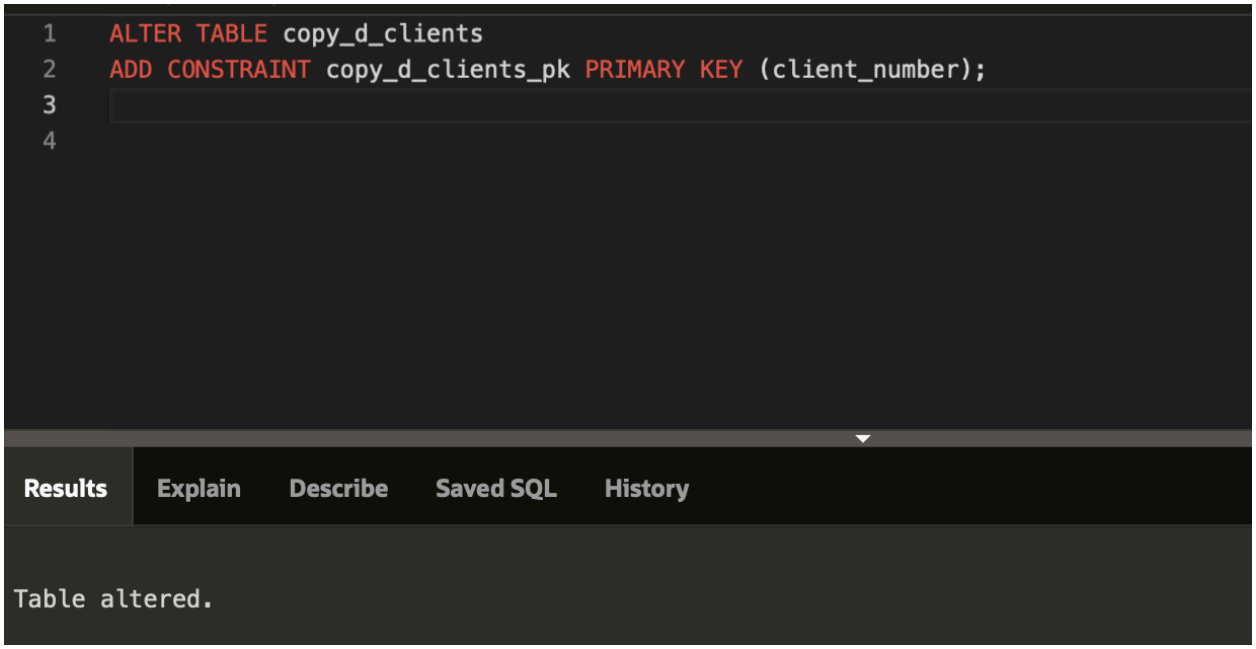
b. ON DELETE SET NULL: When a row in the parent table (adoptions) is deleted, the adoption_id in the child table (animals) will be set to NULL, but the corresponding row in the child table will not be deleted.

7. What are the restrictions on defining a CHECK constraint?

The condition in the CHECK constraint must be a logical expression; it cannot reference other rows or tables, as CHECK constraints only operate on individual rows of the table. A CHECK constraint also cannot contain subqueries and the logic in the CHECK constraint is not enforced on NULL values, so if a column has NULL, it is considered valid.

14-3: Managing Constraints

1. What are four functions that an ALTER statement can perform on constraints?
    - ADD
    - DROP
    - DISABLE
    - ENABLE
2. Since the tables are copies of the original tables, the integrity rules are not passed onto the new tables; only the column datatype definitions remain. You will need to add a PRIMARY KEY constraint to the copy_d_clients table. Name the primary key copy_d_clients_pk . What is the syntax you used to create the PRIMARY KEY constraint to the copy_d_clients.table?



3. Create a FOREIGN KEY constraint in the copy_d_events table. Name the foreign key copy_d_events_fk. This key references the copy_d_clients table client_number column. What is the syntax you used to create the FOREIGN KEY constraint in the copy_d_events table?

```
1  ALTER TABLE copy_d_events
2  ADD CONSTRAINT copy_d_events_fk FOREIGN KEY (client_number)
3  REFERENCES copy_d_clients(client_number);
4
5
```

Results | Explain | Describe | Saved SQL | History

```
Table altered.
```

4. Use a SELECT statement to verify the constraint names for each of the tables. Note that the table names must be capitalized.

    a. The constraint name for the primary key in the copy_d_clients table is _____.

```
1  SELECT constraint_name
2  FROM user_constraints
3  WHERE table_name = 'COPY_D_CLIENTS' AND constraint_type = 'P';
4
5
6
7
8
```

Results | Explain | Describe | Saved SQL | History

CONSTRAINT_NAME
COPY_D_CLIENTS_PK

1 rows returned in 3.59 seconds Download

b. The constraint name for the foreign key in the copy_d_events table is _____.

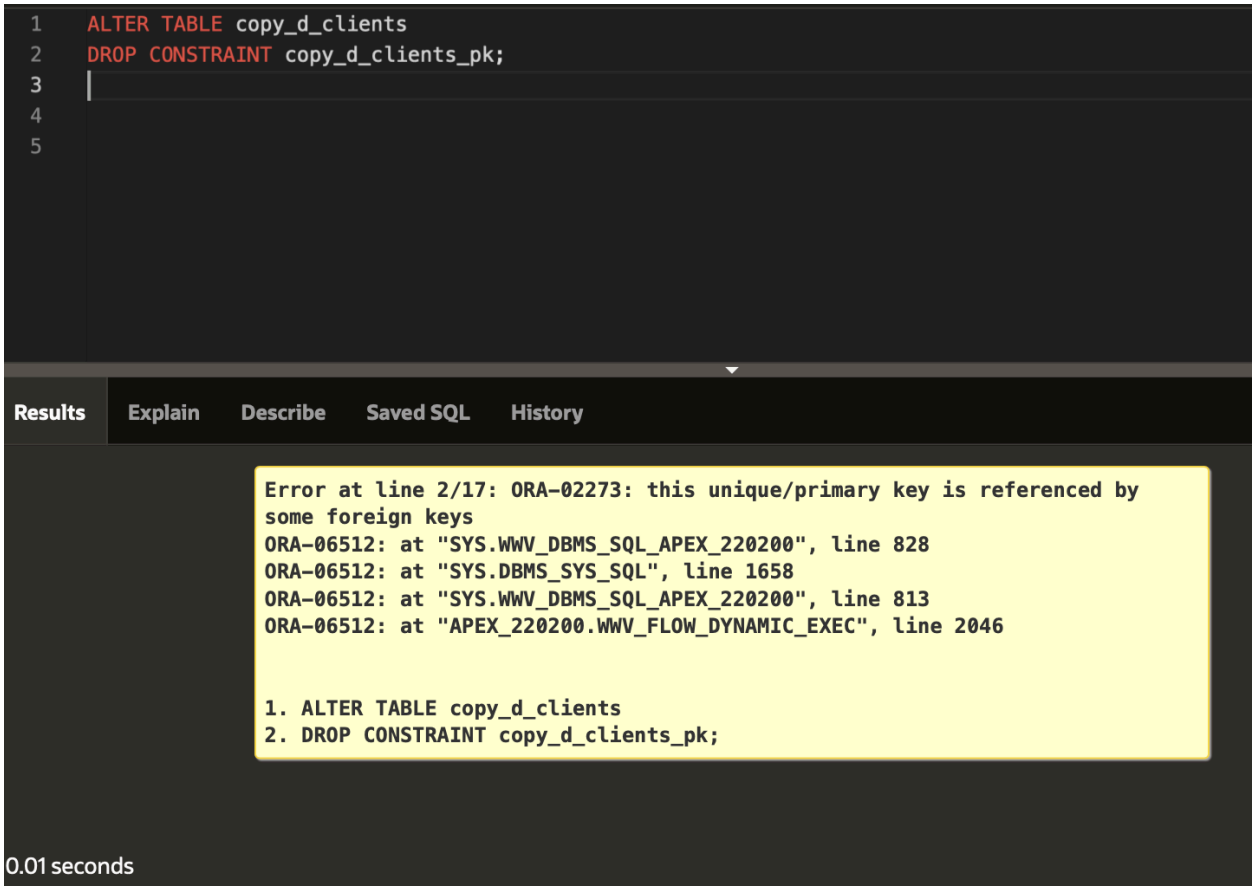
```
1  SELECT constraint_name
2  FROM user_constraints
3  WHERE table_name = 'COPY_D_EVENTS' AND constraint_type = 'R';
4
5
```

Results | Explain | Describe | Saved SQL | History

CONSTRAINT_NAME
COPY_D_EVENTS_FK

1 rows returned in 1.71 seconds Download

5. Drop the PRIMARY KEY constraint on the copy_d_clients table. Explain your results.



You cannot drop a primary key if there are foreign keys referencing it, unless those foreign keys are also removed.

6. Add the following event to the copy_d_events table. Explain your results.

```
1  INSERT INTO copy_d_events (id, name, event_date, description, cost, venue_id, package_code, theme_code, clien
2  VALUES (140, 'Cline Bas Mitzvah', TO_DATE('15-Jul-2004', 'DD-Mon-YYYY'), 'Church and Private Home', 4500, 105
3
4
5
6
```

Results | Explain | Describe | Saved SQL | History

```
ORA-02291: integrity constraint (US_A296_SQL_S36_ADMIN.COPY_D_EVENTS_FK)
violated - parent key not found
ORA-06512: at "SYS.DBMS_SQL", line 1721


1. INSERT INTO copy_d_events (id, name, event_date, description, cost,
venue_id, package_code, theme_code, client_number)
2. VALUES (140, 'Cline Bas Mitzvah', TO_DATE('15-Jul-2004', 'DD-MM-YYYY'),
'Church and Private Home', 4500, 105, 87, 77, 7125);
```

You cannot add the event to the table because there is no parent key found.

7. Create an ALTER TABLE query to disable the primary key in the copy_d_clients table. Then add the values from #6 to the copy_d_events table. Explain your results.

```
1  INSERT INTO copy_d_events (id, name, event_date, description, cost, venue_id, package_code, theme_code, client_
2  VALUES (140, 'Cline Bas Mitzvah', TO_DATE('15-Jul-2004', 'DD-Mon-YYYY'), 'Church and Private Home', 4500, 105,-
3
4
```

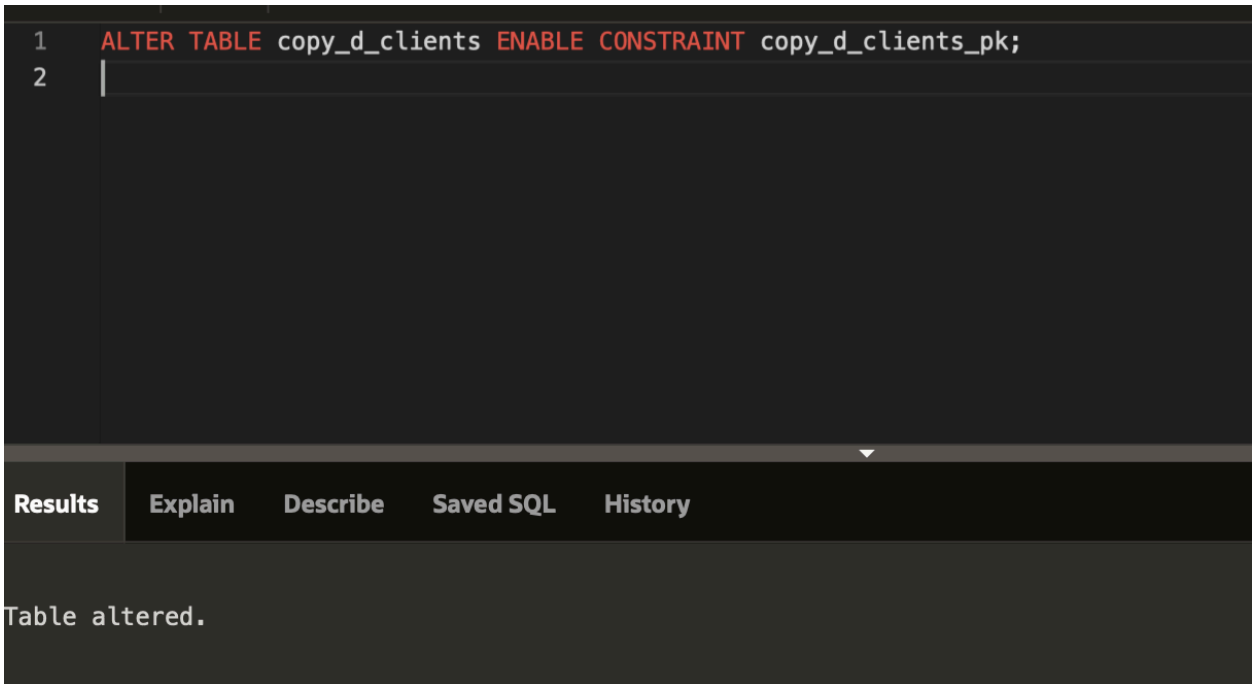
Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

This INSERT statement adds a new row into the copy_d_events table.

8. Repeat question 6: Insert the new values in the copy_d_events table. Explain your results.
This INSERT statement adds a new row into the copy_d_events table.

9. Enable the primary-key constraint in the copy_d_clients table. Explain your results.



Enabling the primary key constraint re-applies the uniqueness and NOT NULL to the client_number column.

10. If you wanted to enable the foreign-key column and reestablish the referential integrity between these two tables, what must be done?
You need to enable the foreign key constraint in the copy_d_events table and make sure that all existing rows in the copy_d_events table have valid client_number values that exist in the copy_d_clients table.

11. Why might you want to disable and then re-enable a constraint?
When loading large amounts of data, disabling constraints temporarily can speed up the process, and re-enabling them ensures integrity after the data load. Disabling and re-enabling a constraint may also improve performance during bulk operations and ensure data consistency.

12. Query the data dictionary for some of the constraints that you have created. How does the data dictionary identify each constraint type?
The data dictionary identifies each constraint type by labeling it: primary key (P) or check (C).

```sql
SELECT constraint_name, constraint_type
FROM user_constraints
WHERE table_name = 'COPY_D_CLIENTS';
```

Results | Explain | Describe | Saved SQL | History

CONSTRAINT_NAME	CONSTRAINT_TYPE
SYS_C0014557705	C
SYS_C0014557706	C
SYS_C0014557707	C
COPY_D_CLIENTS_PK	P

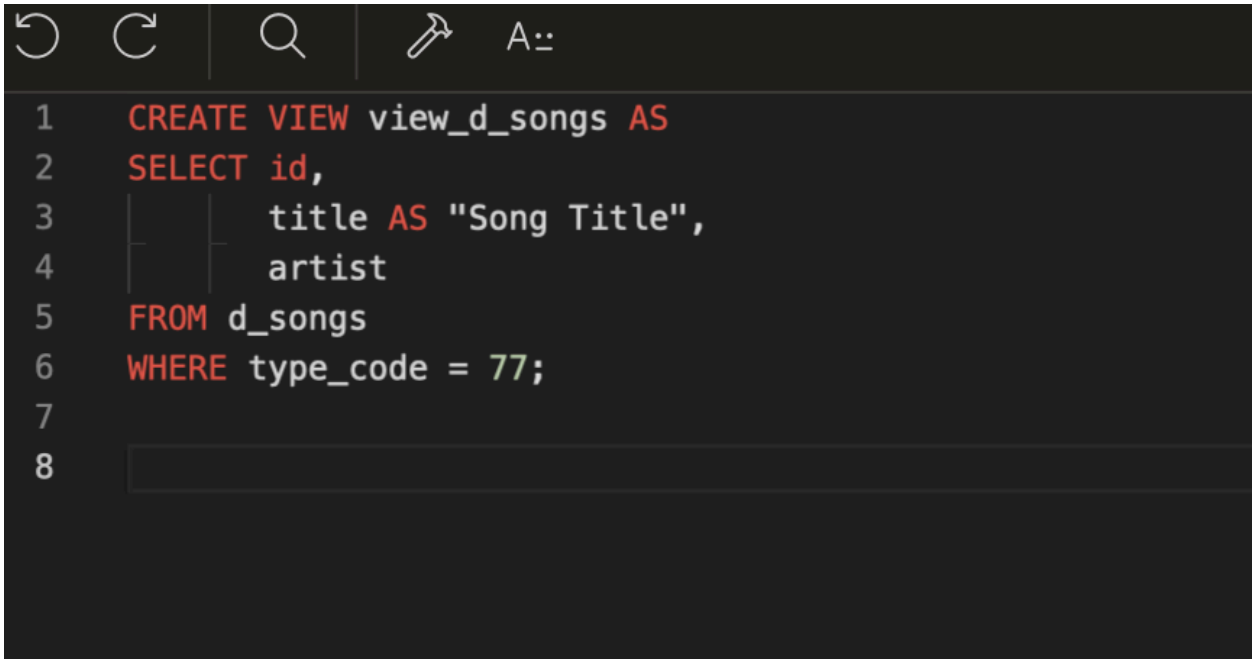
4 rows returned in 1.68 seconds Download

15-1: Creating Views

1. What are three uses for a view from a DBA's perspective?

- Security and data restriction.
- Simplifying complex queries
- Data aggregation and reporting.

2. Create a simple view called view_d_songs that contains the ID, title, and artist from the DJs on Demand table for each "New Age" type code. In the subquery, use the alias "Song Title" for the title column.



```
1  CREATE VIEW view_d_songs AS
2  SELECT id,
3         title AS "Song Title",
4         artist
5  FROM d_songs
6  WHERE type_code = 77;
7
8
```

3. SELECT *
   FROM view_d_songs.
   What was returned?

```
1  SELECT *
2  FROM view_d_songs
3
```

Results | Explain | Describe | Saved SQL | History

ID	Song Title	ARTIST
47	Hurrah for Today	The Jubilant Trio
49	Lets Celebrate	The Celebrants

2 rows returned in 0.00 seconds Download

4. REPLACE view_d_songs. Add type_code to the column list. Use aliases for all columns.

```
1  CREATE OR REPLACE VIEW view_d_songs AS
2  SELECT id AS "Song ID",
3         title AS "Song Title",
4         artist AS "Song Artist",
5         type_code AS "Song Type"
6  FROM d_songs
7  WHERE type_code = 77;
8
9
```

Results | Explain | Describe | Saved SQL | History

```
View created.
```

5. Jason Tsang, the disk jockey for DJs on Demand, needs a list of the past events and those planned for the coming months so he can make arrangements for each event's equipment setup. As the company manager, you do not want him to have access to the price that clients paid for their events. Create a view for Jason to use that

displays the name of the event, the event date, and the theme description. Use aliases for each column name.

```
1   CREATE VIEW view_jason_events AS
2   SELECT name AS "Event Name",
3          event_date AS "Event Date",
4          description AS "Theme Description"
5   FROM d_events
6   WHERE event_date >= SYSDATE;
7
```

Results | Explain | Describe | Saved SQL | History

```
View created.
```

6. It is company policy that only upper-level management be allowed access to individual employee salaries. The department managers, however, need to know the minimum, maximum, and average salaries, grouped by department. Use the Oracle database to prepare a view that displays the needed information for department managers.

```sql
CREATE VIEW view_salary_summary AS
SELECT department_id,
       MIN(salary) AS "Min Salary",
       MAX(salary) AS "Max Salary",
       AVG(salary) AS "Avg Salary"
FROM employees
GROUP BY department_id;

```

Results | Explain | Describe | Saved SQL | History

View created.

15-2: DML Operations and Views

1. Query the data dictionary USER_UPDATABLE_COLUMNS to make sure the columns in the base tables will allow UPDATE, INSERT, or DELETE. Use a SELECT statement. All table names in the data dictionary are stored in uppercase.

```
1  SELECT table_name, column_name, updatable
2  FROM user_updatable_columns
3  WHERE table_name IN ('COPY_D_SONGS', 'COPY_D_EVENTS', 'COPY_D_CDS', 'COPY_D_CLIENTS');
4
5
6
```

Results | Explain | Describe | Saved SQL | History

TABLE_NAME	COLUMN_NAME	UPDATABLE
COPY_D_CDS	CD_NUMBER	YES
COPY_D_CDS	TITLE	YES
COPY_D_CDS	PRODUCER	YES
COPY_D_CDS	YEAR	YES
COPY_D_CLIENTS	CLIENT_NUMBER	YES
COPY_D_CLIENTS	FIRST_NAME	YES
COPY_D_CLIENTS	LAST_NAME	YES
COPY_D_CLIENTS	PHONE	YES
COPY_D_CLIENTS	EMAIL	YES

2. Use the CREATE or REPLACE option to create a view of all the columns in the copy_d_songs table called view_copy_d_songs.

```
1  CREATE OR REPLACE VIEW view_copy_d_songs AS
2  SELECT * FROM copy_d_songs;
3
```

Results | Explain | Describe | Saved SQL | History

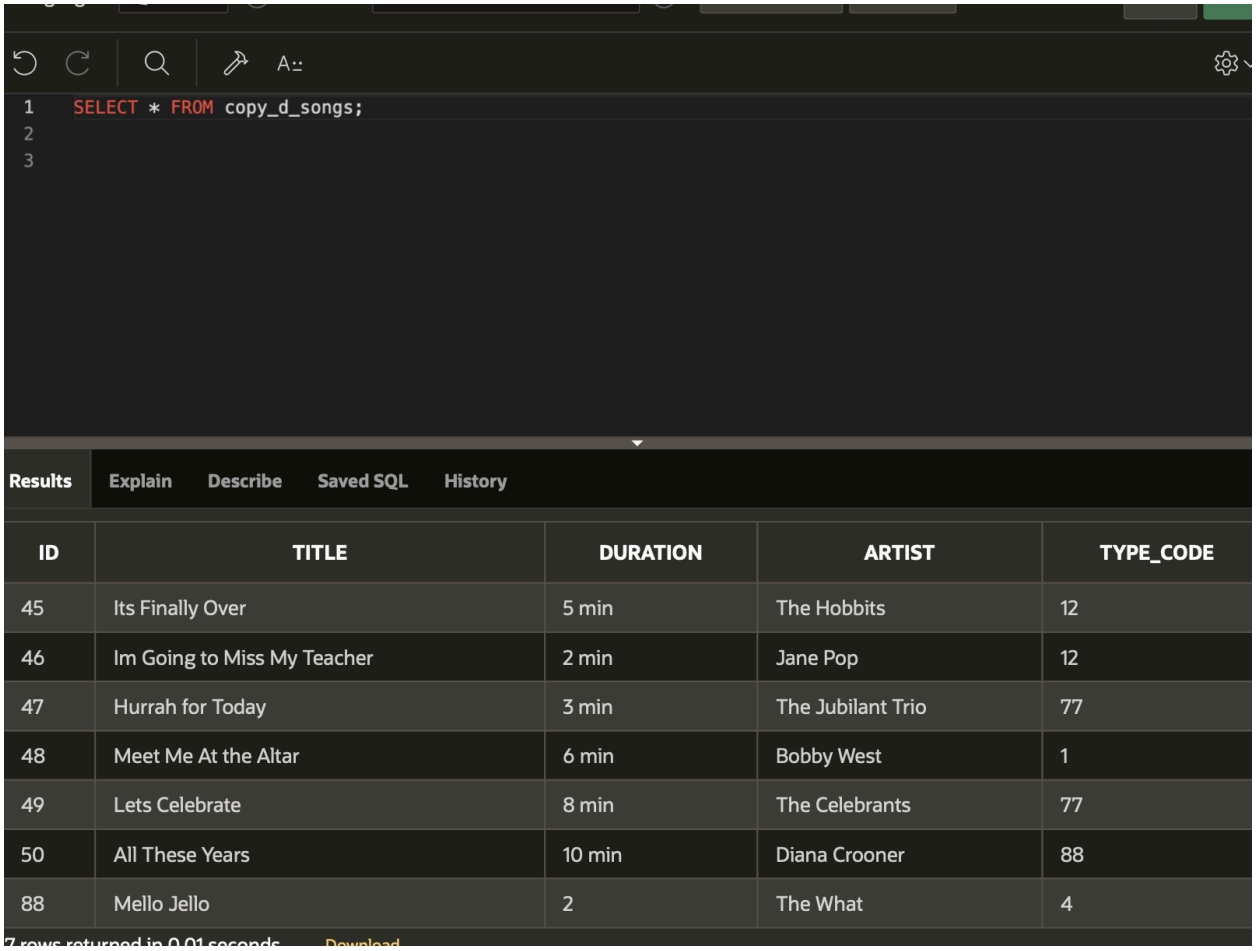
```
View created.
```

3. Use view_copy_d_songs to INSERT the following data into the underlying copy_d_songs table. Execute a SELECT * from copy_d_songs to verify your DML command.

```
1  INSERT INTO view_copy_d_songs (id, title, duration, artist, type_code)
2  VALUES (88, 'Mello Jello', 2, 'The What', 4);
3
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```



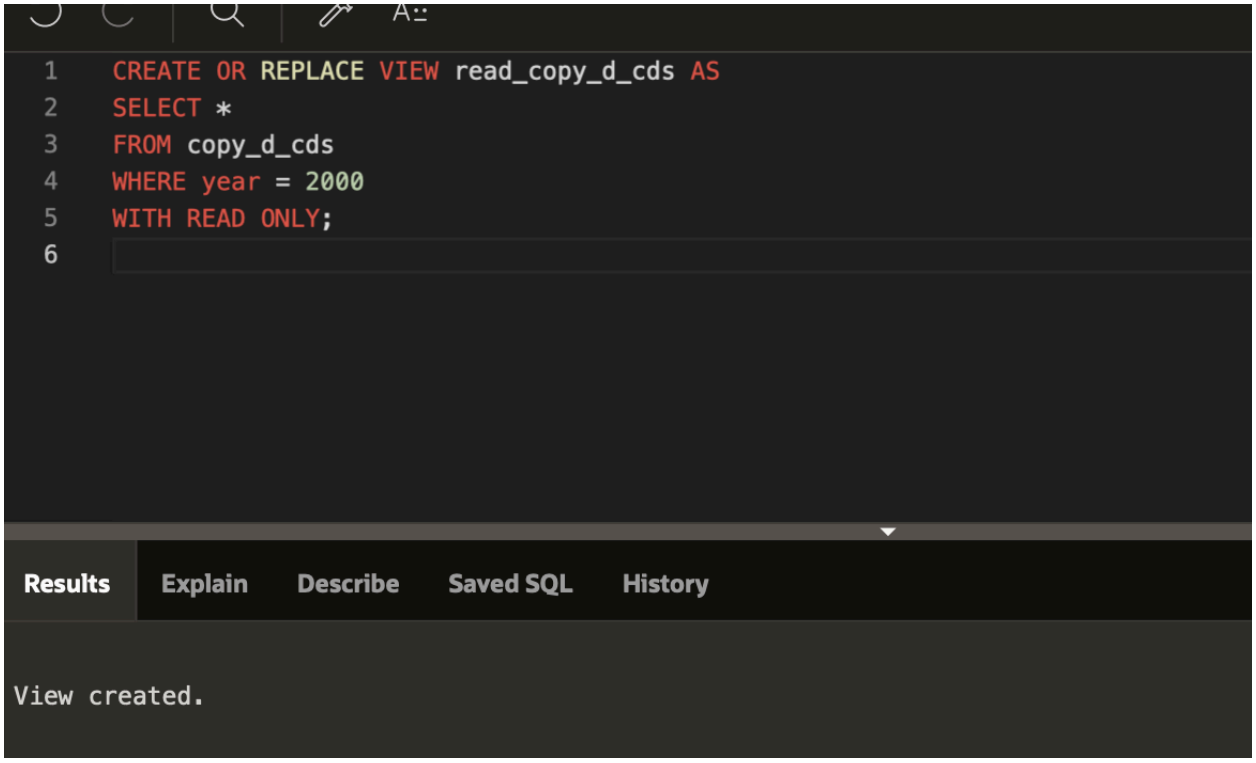
```
SELECT * FROM copy_d_songs;
```

Results | Explain | Describe | Saved SQL | History

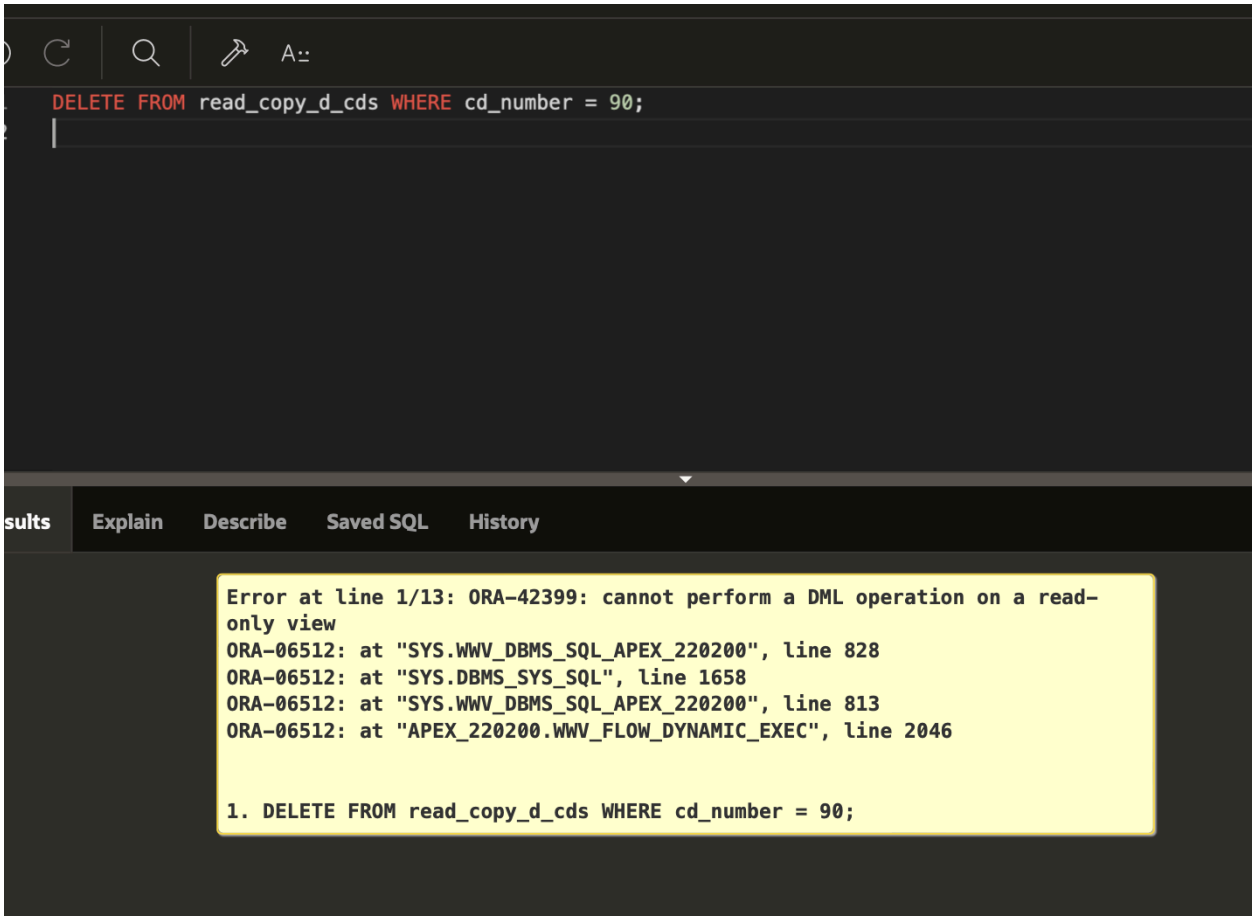
ID	TITLE	DURATION	ARTIST	TYPE_CODE
45	Its Finally Over	5 min	The Hobbits	12
46	Im Going to Miss My Teacher	2 min	Jane Pop	12
47	Hurrah for Today	3 min	The Jubilant Trio	77
48	Meet Me At the Altar	6 min	Bobby West	1
49	Lets Celebrate	8 min	The Celebrants	77
50	All These Years	10 min	Diana Crooner	88
88	Mello Jello	2	The What	4

7 rows returned in 0.01 seconds Download

4. Create a view based on the DJs on Demand COPY_D_CDS table. Name the view read_copy_d_cds. Select all columns to be included in the view. Add a WHERE clause to restrict the year to 2000. Add the WITH READ ONLY option.



5. Using the read_copy_d_cds view, execute a DELETE FROM read_copy_d_cds WHERE cd_number = 90;



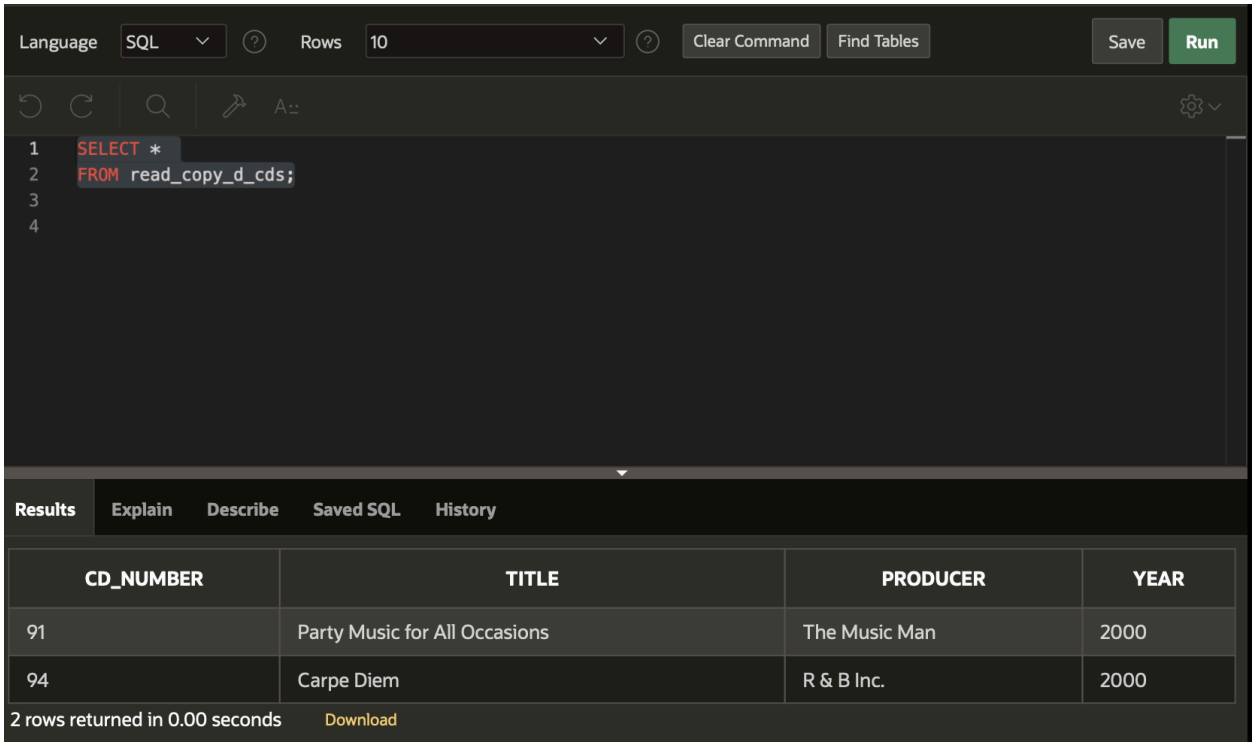
6. Use REPLACE to modify read_copy_d_cds. Replace the READ ONLY option with WITH CHECK OPTION CONSTRAINT ck_read_copy_d_cds. Execute a SELECT * statement to verify that the view exists.

```sql
CREATE OR REPLACE VIEW read_copy_d_cds AS
SELECT *
FROM copy_d_cds
WHERE year = 2000
WITH CHECK OPTION CONSTRAINT ck_read_copy_d_cds;
```

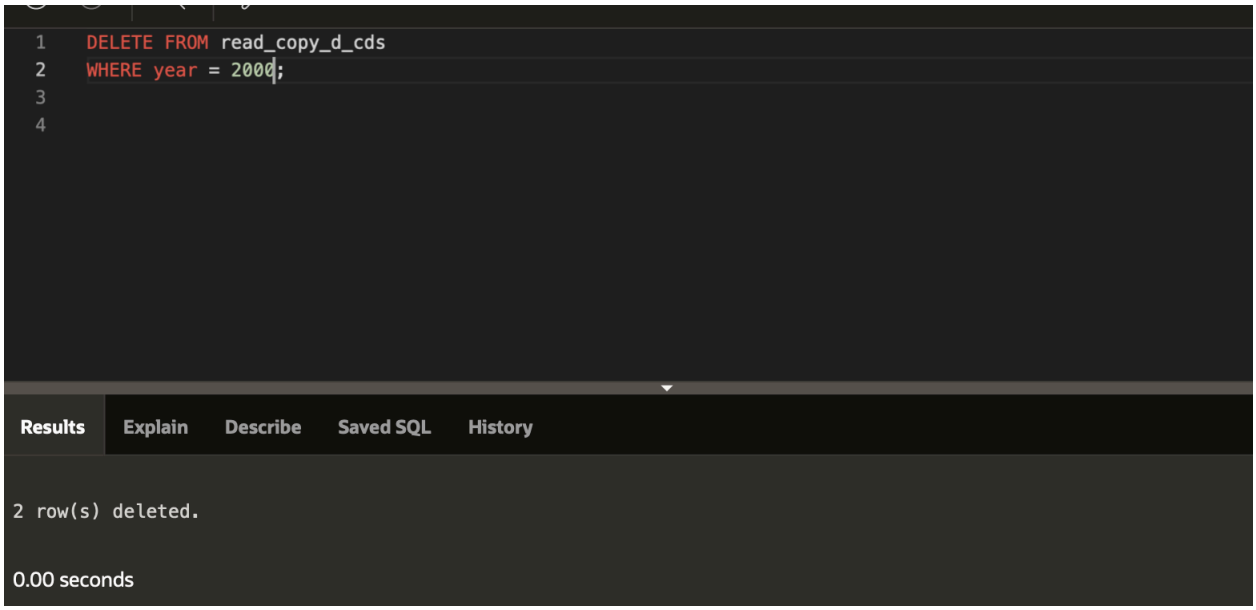
Results | Explain | Describe | Saved SQL | History

```
View created.

0.02 seconds
```



7. Use the read_copy_d_cds view to delete any CD of year 2000 from the underlying copy_d_cds.



8. Use the read_copy_d_cds view to delete cd_number 90 from the underlying copy_d_cds table.

```
1  DELETE FROM read_copy_d_cds
2  WHERE cd_number = 90;
3
4
```

Results | Explain | Describe | Saved SQL | History

```
0 row(s) deleted.
```

9. Use the read_copy_d_cds view to delete year 2001 records.

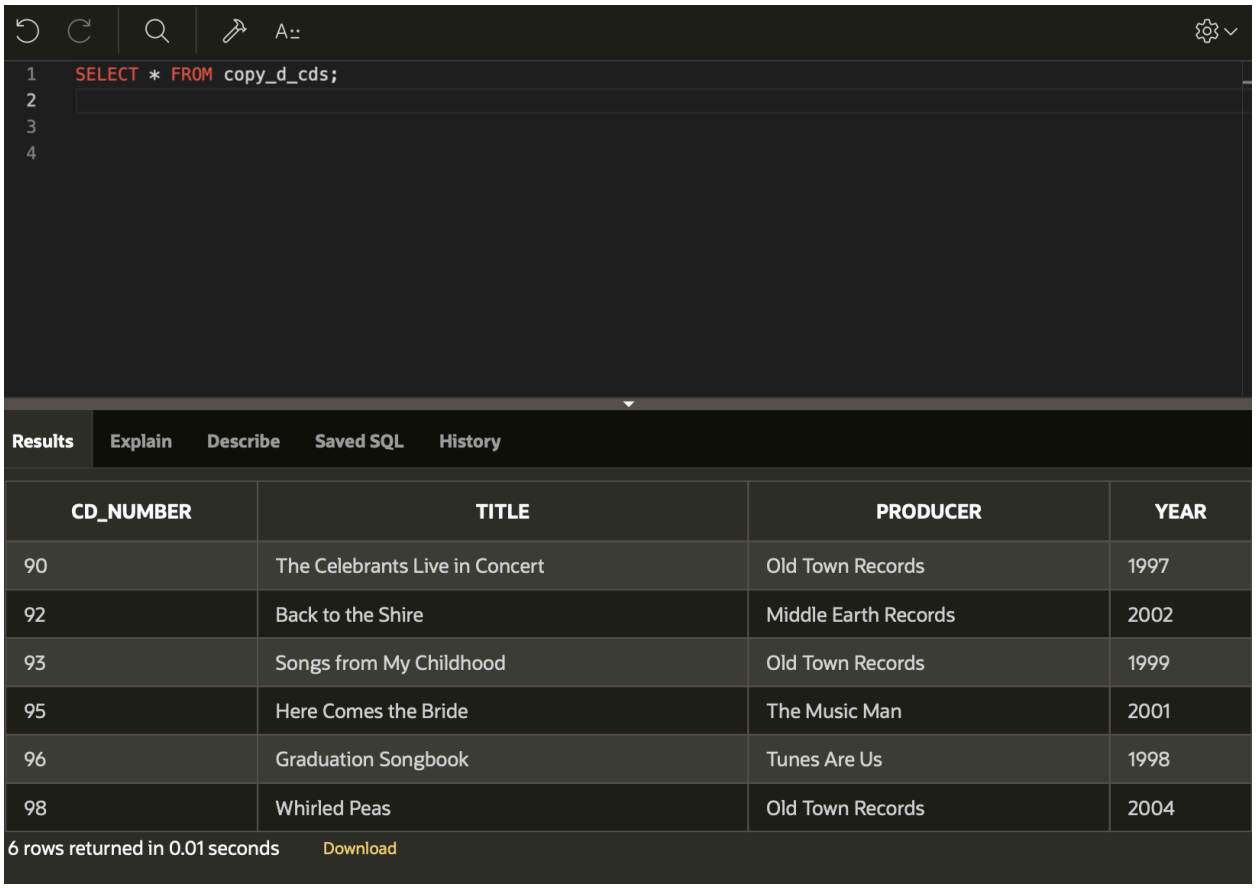
```
1  DELETE FROM read_copy_d_cds
2  WHERE year = 2001;
3
4
5
```

Results | Explain | Describe | Saved SQL | History

```
0 row(s) deleted.

0.00 seconds
```

10. Execute a SELECT * statement for the base table copy_d_cds. What rows were deleted?



```
SELECT * FROM copy_d_cds;
```

Results | Explain | Describe | Saved SQL | History

CD_NUMBER	TITLE	PRODUCER	YEAR
90	The Celebrants Live in Concert	Old Town Records	1997
92	Back to the Shire	Middle Earth Records	2002
93	Songs from My Childhood	Old Town Records	1999
95	Here Comes the Bride	The Music Man	2001
96	Graduation Songbook	Tunes Are Us	1998
98	Whirled Peas	Old Town Records	2004

6 rows returned in 0.01 seconds Download

11. What are the restrictions on modifying data through a view?

Not all views are updatable. For a view to be updatable, it must meet certain conditions. WITH READ ONLY it prevents any modifications to the underlying table through the view and WITH CHECK OPTION it makes sure that only rows that meet the view's WHERE clause can be inserted or updated. Any operation that violates the view's condition is rejected.

12. What is Moore's Law? Do you consider that it will continue to apply indefinitely? Support your opinion with research from the internet.

Moore's Law is the idea that the number of transistors on a computer chip doubles approximately every two years, which leads to faster and cheaper computers over time. However, as transistors get smaller and approach the size of atoms, it becomes harder to keep doubling their number. This means that Moore's Law might not continue indefinitely because of physical limits. To keep improving technology, scientists and engineers are exploring new ideas, like 3D chips or quantum computing, which could help maintain progress even after Moore's

Law slows down. In short, while Moore's Law has been a reliable guide for technological growth, it may not hold forever, and future advances may rely on new technologies.

https://cap.csail.mit.edu/death-moores-law-what-it-means-and-what-might-fill-gap-going-forward

13. What is the “singularity” in terms of computing?

In terms of computing, the singularity refers to the hypothetical future point where technological growth becomes uncontrollable and irreversible, leading to unforeseeable changes in human civilization.

15-3: Managing Views

1. Create a view from the copy_d_songs table called view_copy_d_songs that includes only the title and artist. Execute a SELECT * statement to verify that the view exists.

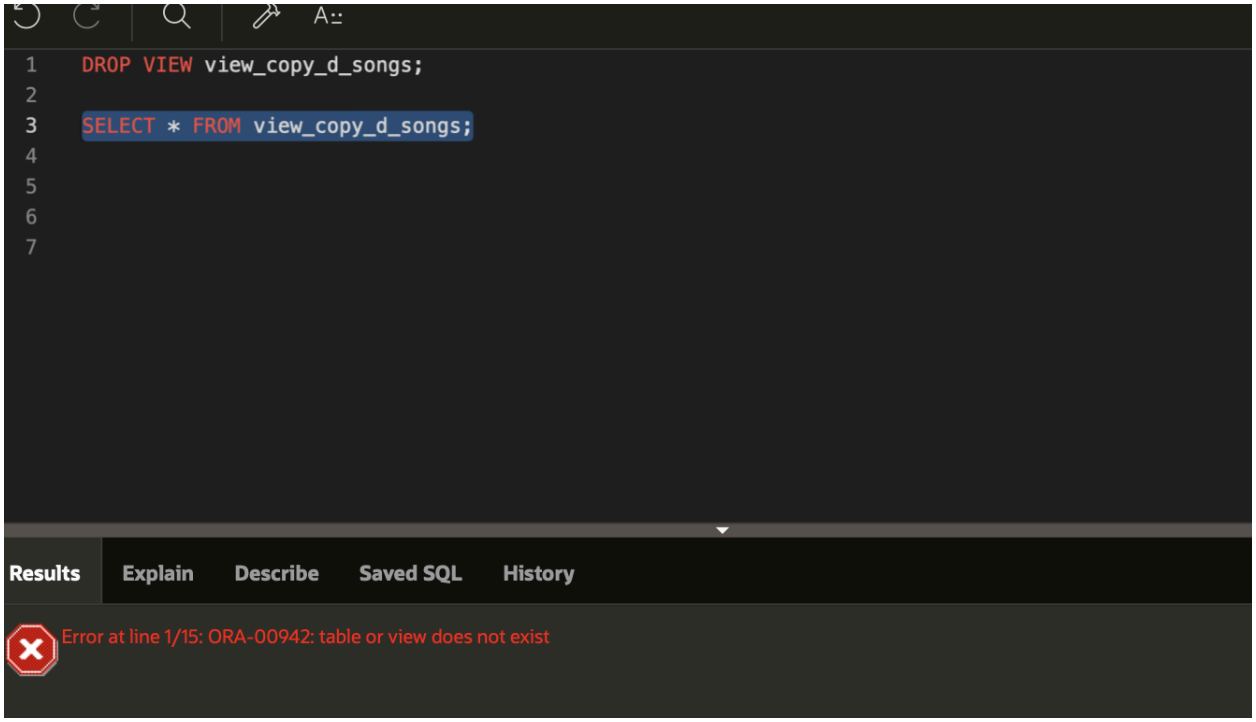
```
1  CREATE VIEW view_copy_d_songs AS
2  SELECT title, artist
3  FROM copy_d_songs;
4
5  SELECT * FROM view_copy_d_songs;
6
7
8
9
```

Results | Explain | Describe | Saved SQL | History

ID	TITLE	DURATION	ARTIST	
45	Its Finally Over	5 min	The Hobbits	12
46	Im Going to Miss My Teacher	2 min	Jane Pop	12
47	Hurrah for Today	3 min	The Jubilant Trio	77
48	Meet Me At the Altar	6 min	Bobby West	1
49	Lets Celebrate	8 min	The Celebrants	77
50	All These Years	10 min	Diana Crooner	88
88	Mello Jello	2	The What	4

7 rows returned in 0.00 seconds Download

2. Issue a DROP view_copy_d_songs. Execute a SELECT * statement to verify that the view has been deleted.



3. Create a query that selects the last name and salary from the Oracle database. Rank the salaries from highest to lowest for the top three employees.

```
1  SELECT last_name, salary
2  FROM employees
3  ORDER BY salary DESC
4  FETCH FIRST 3 ROWS ONLY;
5
6
```

Results | Explain | Describe | Saved SQL | History

LAST_NAME	
King	24000
Kochhar	17000
De Haan	17000

rows returned in 0.00 seconds Download

4. Construct an inline view from the Oracle database that lists the last name, salary, department ID, and maximum salary for each department. Hint: One query will need to calculate maximum salary by department ID.

```sql
SELECT last_name, salary, department_id,
       (SELECT MAX(salary) FROM employees WHERE department_id = e.department_id) AS max_salary
FROM employees e;
```

Results | Explain | Describe | Saved SQL | History

LAST_NAME	SALARY	DEPARTMENT_ID	MAX_SALARY
Mourgos	5800	50	5800
Rajs	3500	50	5800
Davies	3100	50	5800
Matos	2600	50	5800
Vargas	2500	50	5800
Bell	3500	50	5800
Heiden	2600	50	5800
Higgins	12000	110	12000

5. Create a query that will return the staff members of Global Fast Foods ranked by salary from lowest to highest.

```
1  SELECT * FROM f_staffs
2  ORDER BY salary ASC;
3
4
```

Results | Explain | Describe | Saved SQL | History

ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING	STAFF_TYPE	MANAGE
12	Sue	Doe	01-Jul-1980	6.75	10.25	-	Order Taker	19
9	Bob	Miller	19-Mar-1979	10	-	Grill	Cook	19
19	Monique	Tuttle	30-Mar-1969	60	-	-	Manager	-

rows returned in 0.01 seconds Download

Extension Exercises:

1.

```
1  CREATE TABLE my_departments AS
2  SELECT * FROM departments;
3
4  SELECT * FROM my_departments;
5
```

Results | Explain | Describe | Saved SQL | History

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales - Europe	149	2500
85	Sales - Americas	149	2100
90	Executive	100	1700

2.

```
1  DESCRIBE my_departments;
2
3
4
```

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **MY_DEPARTMENTS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Commer
MY_DEPARTMENTS	DEPARTMENT_ID	NUMBER	-	4	0	-	✓	-	-
	DEPARTMENT_NAME	VARCHAR2	30	-	-	-	-	-	-
	MANAGER_ID	NUMBER	-	6	0	-	✓	-	-
	LOCATION_ID	NUMBER	-	4	0	-	✓	-	-

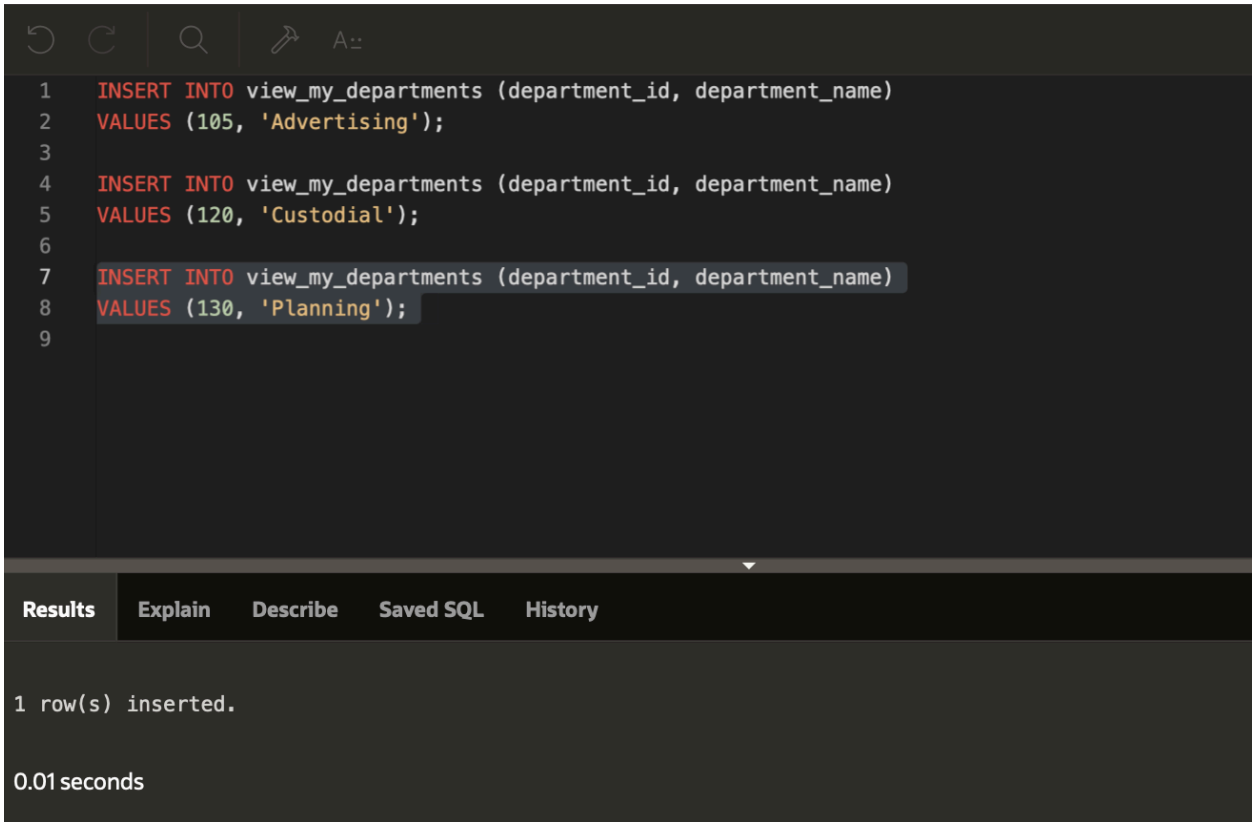
3.

```
1  CREATE VIEW view_my_departments AS
2  SELECT department_id, department_name
3  FROM my_departments;
4
5
6
7
```

Results | Explain | Describe | Saved SQL | History

```
View created.
```

4.

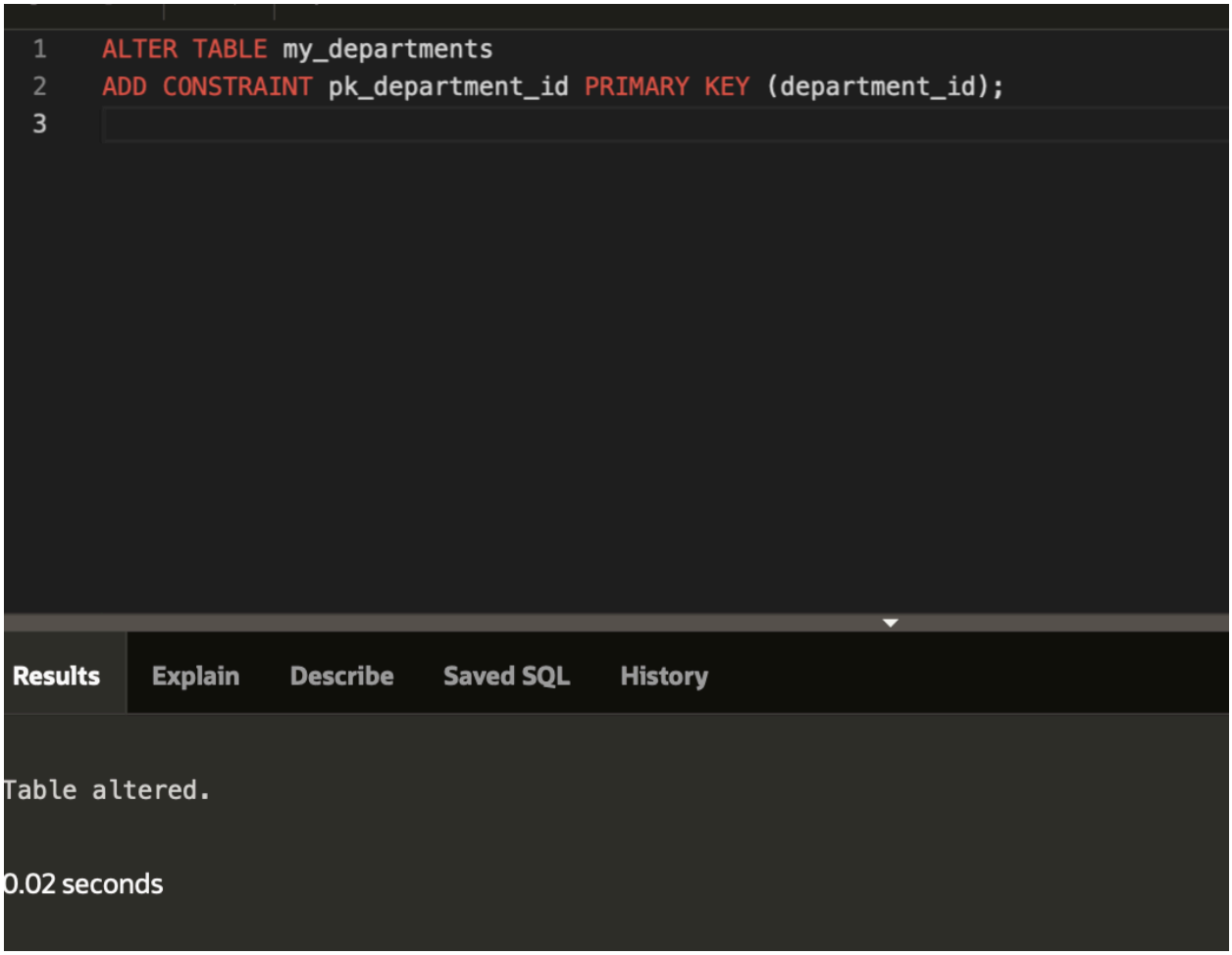


```
1  INSERT INTO view_my_departments (department_id, department_name)
2  VALUES (105, 'Advertising');
3
4  INSERT INTO view_my_departments (department_id, department_name)
5  VALUES (120, 'Custodial');
6
7  INSERT INTO view_my_departments (department_id, department_name)
8  VALUES (130, 'Planning');
9
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

0.01 seconds

5. 

6. INSERT INTO view_my_departments (department_name)
   VALUES ('Human Resources');

7.

```
1  INSERT INTO view_my_departments (department_id, department_name)
2  VALUES (220, 'Human Resources');
3
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

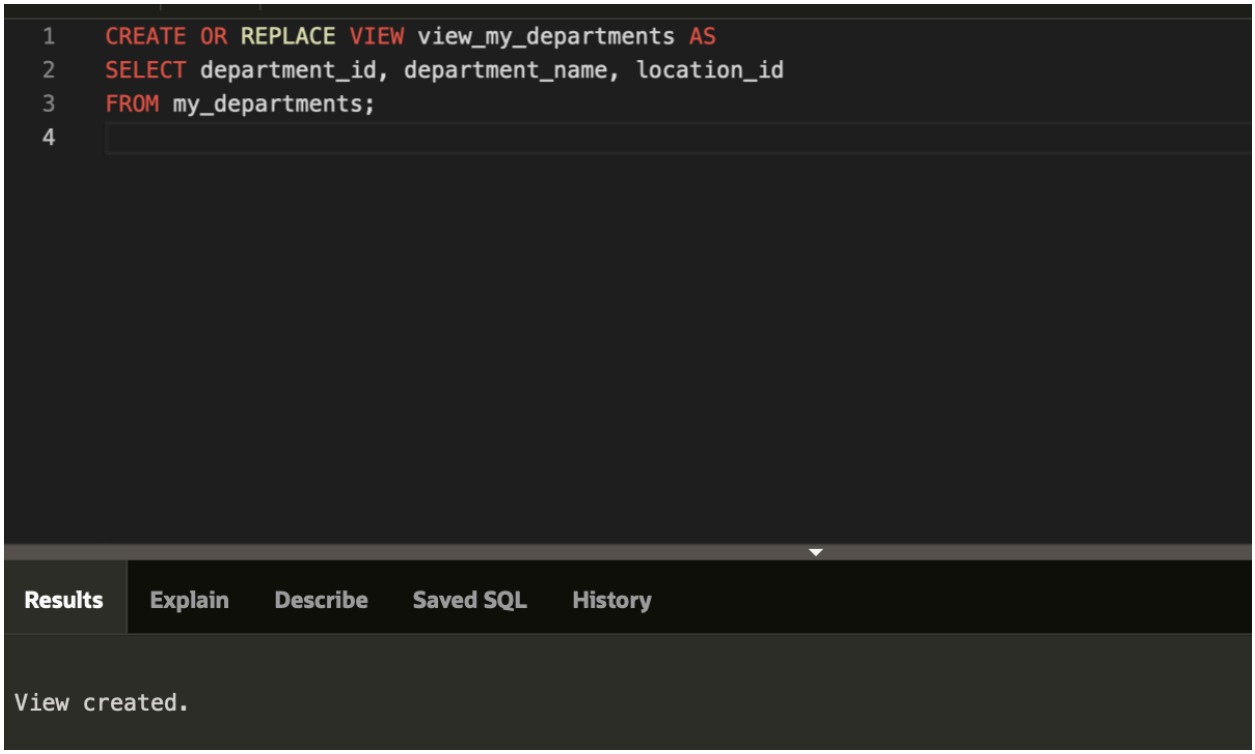
8.

```
1  SELECT * FROM view_my_departments;
2
```

esults | Explain | Describe | Saved SQL | History

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
50	Shipping
60	IT
80	Sales - Europe
85	Sales - Americas
90	Executive
110	Accounting

9.

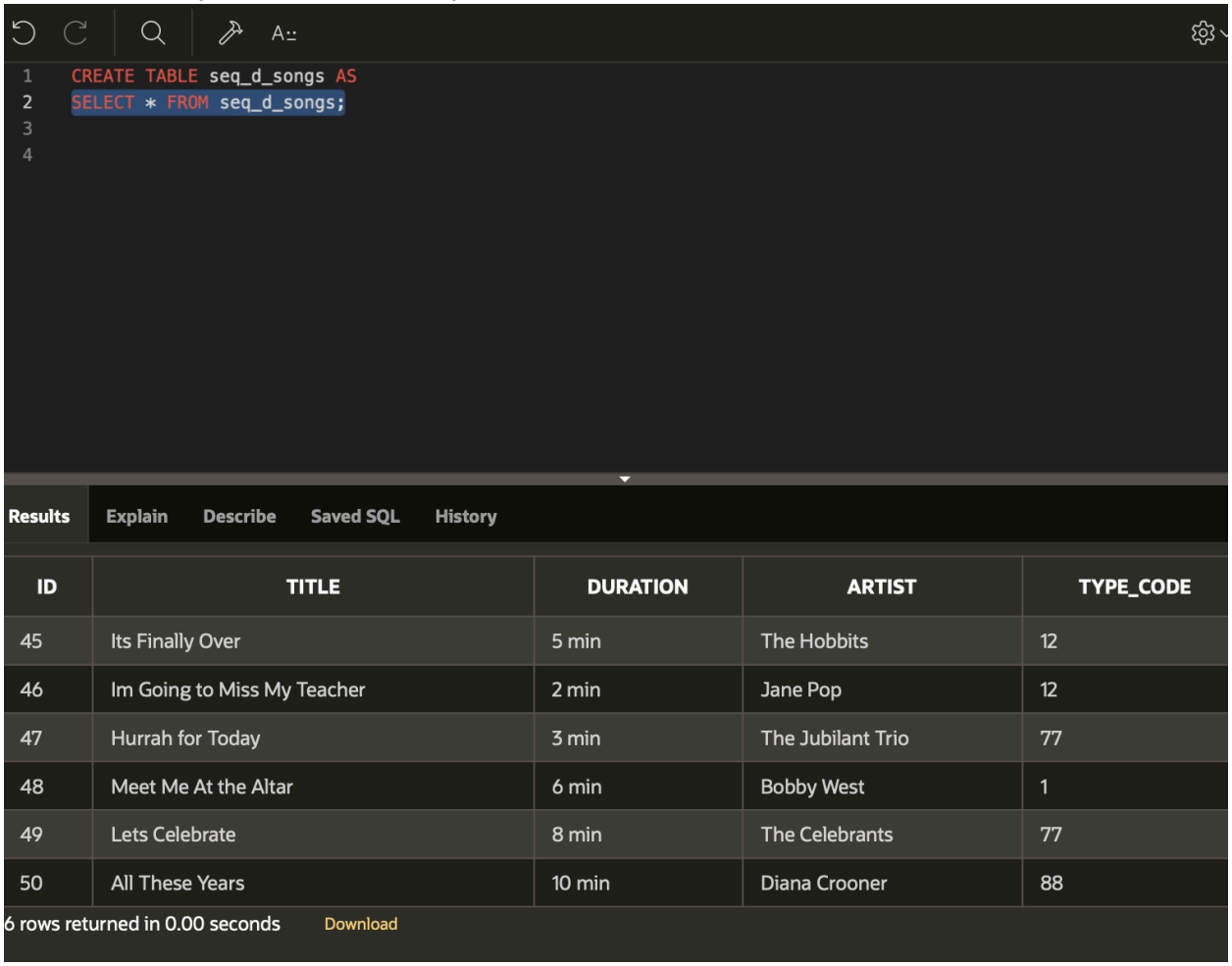


10.
```
ALTER TABLE my_departments
MODIFY location_id NOT NULL;
```

11.
```
CREATE VIEW view_location_departments AS
SELECT d.department_name, l.street_address, l.city, l.state
FROM locations l
JOIN departments d ON l.location_id = d.location_id
WHERE l.city IN ('U.S. City Names');
```

16-1: Working with Sequences

1. Using CREATE TABLE AS subquery syntax, create a seq_d_songs table of all the columns in the DJs on Demand database table d_songs. Use the SELECT * in the subquery to make sure that you have copied all of the columns.



```
1  CREATE TABLE seq_d_songs AS
2  SELECT * FROM seq_d_songs;
3
4
```

Results | Explain | Describe | Saved SQL | History

ID	TITLE	DURATION	ARTIST	TYPE_CODE
45	Its Finally Over	5 min	The Hobbits	12
46	Im Going to Miss My Teacher	2 min	Jane Pop	12
47	Hurrah for Today	3 min	The Jubilant Trio	77
48	Meet Me At the Altar	6 min	Bobby West	1
49	Lets Celebrate	8 min	The Celebrants	77
50	All These Years	10 min	Diana Crooner	88

6 rows returned in 0.00 seconds Download

2. Because you are using copies of the original tables, the only constraints that were carried over were the NOT NULL constraints. Create a sequence to be used with the primary-key column of the seq_d_songs table. To avoid assigning primary-key numbers to these tables that already exist, the sequence should start at 100 and have a maximum

value of 1000. Have your sequence increment by 2 and have NOCACHE and NOCYCLE. Name the sequence seq_d_songs_seq.

```
1  CREATE SEQUENCE seq_d_songs_seq
2  START WITH 100
3  INCREMENT BY 2
4  MAXVALUE 1000
5  NOCACHE
6  NOCYCLE;
7
8
```

Results | Explain | Describe | Saved SQL | History

```
Sequence created.
```

3. Query the USER_SEQUENCES data dictionary to verify the seq_d_songs_seq SEQUENCE settings.

```
1  SELECT sequence_name
2  FROM user_sequences
3  WHERE sequence_name = 'SEQ_D_SONGS_SEQ';
4
5
```

Results | Explain | Describe | Saved SQL | History

SEQUENCE_NAME
SEQ_D_SONGS_SEQ

1 rows returned in 0.01 seconds Download

4. Insert two rows into the seq_d_songs table. Be sure to use the sequence that you created for the ID column. Add the two songs shown in the graphic.

```
1  INSERT INTO seq_d_songs (id, title, duration, artist, type_code)
2  VALUES (seq_d_songs_seq.NEXTVAL, 'Island Fever', '5 min', 'Hawaiian Islanders', 12);
3
4  INSERT INTO seq_d_songs (id, title, duration, artist, type_code)
5  VALUES (seq_d_songs_seq.NEXTVAL, 'Castle of Dreams', '4 min', 'The Wanderers', 77);
6
7
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

5. Write out the syntax for seq_d_songs_seq to view the current value for the sequence. Use the DUAL table. (Oracle Application Developer will not run this query.)

SELECT seq_d_songs_seq.CURRVAL FROM dual;

6. What are three benefits of using SEQUENCEs?

Sequences provide unique numbers, ensuring that primary key values do not repeat. Sequences can also have different values to meet various application needs and are generated fairly quickly.

7. What are the advantages of caching sequence values?

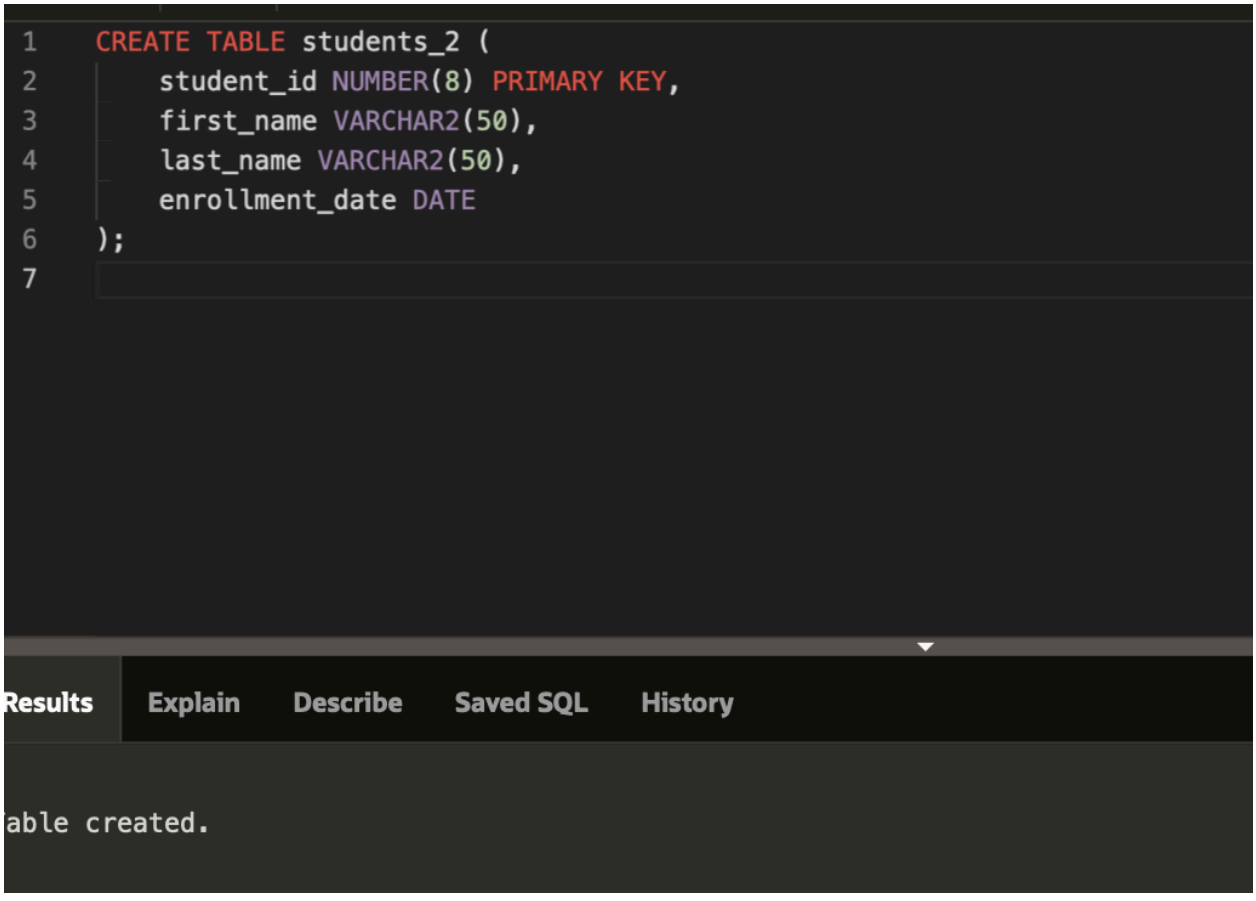
Caching sequence values can reduce the number of operations needed to retrieve sequence values.

8. Name three reasons why gaps may occur in a sequence?

Some reasons why gaps may occur in a sequence is If a roll back applies, the number generated during that transaction can be lost. Another reason is if a sequence is cached and the system crashes, those values can no longer be used. Lastly, when NEXTVAL is called for generating sequence values and the generated value is not used, it can cause gaps.

Extension Exercises:

1.



2.

```
1   CREATE SEQUENCE student_id_seq
2   START WITH 1
3   INCREMENT BY 1
4   MAXVALUE 99999999
5   NOCACHE
6   NOCYCLE;
7
```

Results | Explain | Describe | Saved SQL | History

```
Sequence created.
```

3.

```
INSERT INTO students_2 (student_id, first_name, last_name, enrollment_date)
VALUES (student_id_seq.NEXTVAL, 'John', 'Doe', SYSDATE);

INSERT INTO students_2 (student_id, first_name, last_name, enrollment_date)
VALUES (student_id_seq.NEXTVAL, 'Jane', 'Smith', SYSDATE);

SELECT * FROM students_2
```

Results | Explain | Describe | Saved SQL | History

STUDENT_ID	FIRST_NAME	LAST_NAME	ENROLLMENT_DATE
2	Jane	Smith	02-Dec-2024
1	John	Doe	02-Dec-2024

2 rows returned in 0.01 seconds Download

16-2: Indexes and Synonyms

1. What is an index and what is it used for?

An index is a database object that improves the speed of data retrieval operations on a table. Indexes are mainly used to speed up query performance, especially for columns frequently involved in WHERE, ORDER BY, JOIN, and other operations.

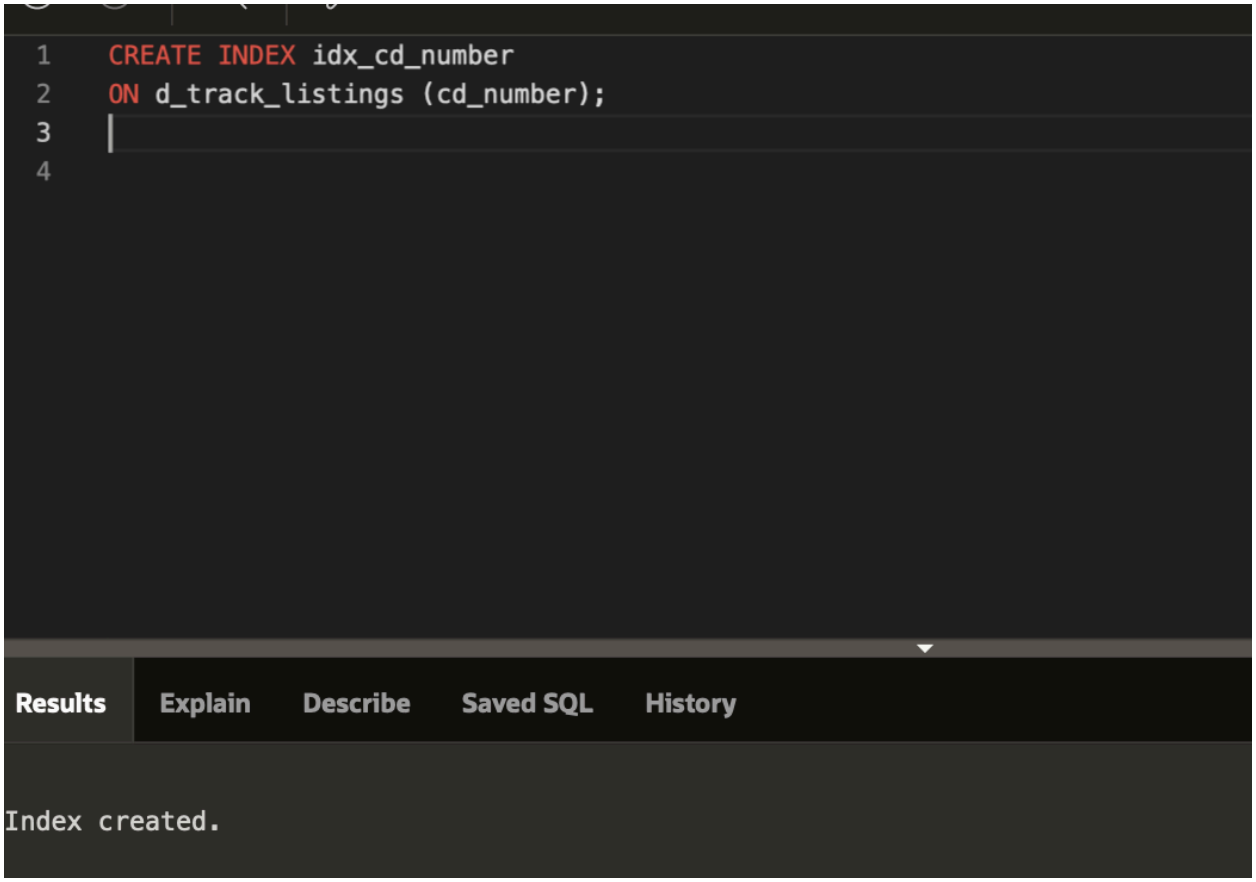
2. What is a ROWID, and how is it used?

A ROWID is a unique identifier for each row in a table. It represents the physical location of a row in a database.

3. When will an index be created automatically?

When a primary key or unique key constraint is defined on a table and when a unique constraint or primary key is added to a column.

4. Create a nonunique index (foreign key) for the DJs on Demand column (cd_number) in the D_TRACK_LISTINGS table. Use the Oracle Application Developer SQL Workshop Data Browser to confirm that the index was created.



5. Use the join statement to display the indexes and uniqueness that exist in the data dictionary for the DJs on Demand D_SONGS table.

```sql
SELECT ui.index_name, ui.uniqueness
FROM user_indexes ui
JOIN user_tab_columns utc ON ui.table_name = utc.table_name
WHERE ui.table_name = 'D_SONGS';
```

Results | Explain | Describe | Saved SQL | History

INDEX_NAME	UNIQUENESS
D_SNG_ID_PK	UNIQUE
D_SNG_ID_PK	UNIQUE
D_SNG_ID_PK	UNIQUE
D_SNG_ID_PK	UNIQUE
D_SNG_ID_PK	UNIQUE

5 rows returned in 1.47 seconds Download

6. Use a SELECT statement to display the index_name, table_name, and uniqueness from the data dictionary USER_INDEXES for the DJs on Demand D_EVENTS table.

```
1  SELECT index_name, table_name, uniqueness
2  FROM user_indexes
3  WHERE table_name = 'D_EVENTS';
4
```

Results | Explain | Describe | Saved SQL | History

INDEX_NAME	TABLE_NAME	UNIQUENESS
D_EVE_ID_PK	D_EVENTS	UNIQUE

rows returned in 0.04 seconds Download

7. Write a query to create a synonym called dj_tracks for the DJs on Demand d_track_listings table.

```
1  CREATE SYNONYM dj_tracks FOR d_track_listings;
2
```

Results | Explain | Describe | Saved SQL | History

```
Synonym created.
```

8. Create a function-based index for the last_name column in DJs on Demand D_PARTNERS table that makes it possible not to have to capitalize the table name for searches. Write a SELECT statement that would use this index.

```sql
CREATE INDEX idx_last_name_lower
ON d_partners (LOWER(last_name));

SELECT * FROM d_partners
WHERE LOWER(last_name) = 'plumb';
```

Results | Explain | Describe | Saved SQL | History

ID	FIRST_NAME	LAST_NAME	EXPERTISE	SPECIALTY	AUTH_EXPENSE_AMT	MANAGER_ID	PARTNER_TYPE
33	Allison	Plumb	Event Planning	-	300000	33	Manager

1 rows returned in 0.01 seconds Download

9. Create a synonym for the D_TRACK_LISTINGS table. Confirm that it has been created by querying the data dictionary.

```sql
CREATE SYNONYM track_listings FOR d_track_listings;

SELECT * FROM user_synonyms
WHERE synonym_name = 'TRACK_LISTINGS';
```

Results | Explain | Describe | Saved SQL | History

SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK	ORIGIN_CON_
TRACK_LISTINGS	US_A296_SQL_S36_ADMIN	D_TRACK_LISTINGS	-	3

1 rows returned in 0.05 seconds Download

10. Drop the synonym that you created in question 9.

```
1  DROP SYNONYM track_listings;
2
```

Results | Explain | Describe | Saved SQL | History

```
Synonym dropped.

0.01 seconds
```

17-1: Controlling User Access

1. What are system privileges concerned with?

System privileges enable users to manage the database environment itself, including creating, altering, or dropping objects like tables, views, or users, and managing other database-level operations.

2. What are object privileges concerned with?

Object privileges are concerned with controlling access to specific database objects like tables, views, or procedures.

3. What is another name for object security?

Object-level security.

4. What commands are necessary to allow Scott access to the database with a password of tiger?

CREATE USER scott IDENTIFIED BY tiger;
GRANT CONNECT TO scott;

5. What are the commands to allow Scott to SELECT from and UPDATE the d_clients table?

GRANT SELECT, UPDATE ON d_clients TO scott;

6. What is the command to allow everybody the ability to view the d_songs table?

GRANT SELECT ON d_songs TO PUBLIC;

7. Query the data dictionary to view the object privileges granted to you the user.

```
1  SELECT * FROM user_tab_privs;
2
```

Results | Explain | Describe | Saved SQL | History

GRANTEE	OWNER	TABLE_NAME	GRANTOR	PRIVILEGE	GRANTABLE	HIERA
PUBLIC	SYS	US_A296_SQL_S36_ADMIN	US_A296_SQL_S36_ADMIN	INHERIT PRIVILEGES	NO	NO

1 rows returned in 3.64 seconds Download

8. What privilege should a user be given to create tables?

GRANT CREATE TABLE TO username;

9. If you create a table, how can you pass along privileges to other users just to view your table?

GRANT SELECT ON your_table TO other_user;

10. What syntax would you use to grant another user access to your copy_employees table?

GRANT SELECT, INSERT, UPDATE ON copy_employees TO another_user;

11. How can you find out what privileges you have been granted for columns in the tables belonging to others?

SELECT * FROM all_tab_privs
WHERE grantee = 'your_username';

17-2: Creating and Revoking Object Privileges

1. What is a role?

A named collection of privileges that can be granted to users or other roles.

2. What are the advantages of a role to a DBA?

Roles make it easier to manage permissions, saves time, reduces errors, flexibility in managing permissions across different levels, and ensures that users only get necessary permissions.

3. Give the ability to another user in your class to look at one of your tables. Give him the right to let other students have that ability.

GRANT SELECT ON my_table TO student2 WITH GRANT OPTION;

4. You are the DBA. You are creating many users who require the same system privileges. What should you use to make your job easier?

I would create roles and assign the required system privileges to these roles. Then, I'd assign the roles to users. This would allow me to manage privileges efficiently.

CREATE ROLE role_name;
GRANT system_privileges TO role_name;
GRANT role_name TO user_name;

5. What is the syntax to accomplish the following?

    a. Create a role of manager that has the privileges to select, insert, and update and delete from the employees table

CREATE ROLE manager;
GRANT SELECT, INSERT, UPDATE, DELETE ON employees TO manager;

    b. Create a role of clerk that just has the privileges of select and insert on the employees table

CREATE ROLE clerk;
GRANT SELECT, INSERT ON employees TO clerk;

    c. Grant the manager role to user scott

GRANT manager TO scott;

    d. Revoke the ability to delete from the employees table from the manager role

REVOKE DELETE ON employees FROM manager;

6. What is the purpose of a database link?
A database link is used to connect and allow communication between two separate databases.

17-3: Regular Expressions

1. Working with the employees table, and using regular expressions, write a query that returns employees whose first names start with a “S” (uppercase) followed by either a “t” (lowercase) or “h” (lowercase).

```
1  SELECT *
2  FROM employees
3  WHERE REGEXP_LIKE(first_name, '^S[t|h]');
4
```

Results | Explain | Describe | Saved SQL | History

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	CO
100	Steven	King	SKING	515.123.4567	17-Jun-2002	AD_PRES	24000	-
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-Jun-2009	AC_MGR	12000	-

2 rows returned in 0.01 seconds Download

2. Investigate the LOCATIONS table.

a. Describe the table.

```
1  DESCRIBE locations;
2
```

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **LOCATIONS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
LOCATIONS	LOCATION_ID	NUMBER	-	4	0	1	-	-	-
	STREET_ADDRESS	VARCHAR2	40	-	-	-	✓	-	-
	POSTAL_CODE	VARCHAR2	12	-	-	-	✓	-	-
	CITY	VARCHAR2	30	-	-	-	-	-	-
	STATE_PROVINCE	VARCHAR2	25	-	-	-	✓	-	-

b. Perform a select that returns all rows and all columns of that table.

```
1  SELECT * FROM locations;
2
3
```

Results | Explain | Describe | Saved SQL | History

LOCATION_ID	STREET_ADDRESS	POSTAL_CODE	CITY	STATE_PROVINCE	COUNTRY_ID
1800	460 Bloor St. W.	ON M5S 1X8	Toronto	Ontario	2
2500	Magdalen Centre, The Oxford Science Park	OX9 9ZB	Oxford	Oxford	44
1400	2014 Jabberwocky Rd	26192	Southlake	Texas	1
1500	2011 Interiors Blvd	99236	South San Francisco	California	1
1700	2004 Charade Rd	98199	Seattle	Washington	1
2100	Av. Rio Branco	20090-003	Rio de Janeiro	Rio de Janeiro	55

6 rows returned in 0.01 seconds Download

c. Write a query using regular expressions that removes the spaces in the street_address column in the LOCATIONS table.

```
1  SELECT REGEXP_REPLACE(street_address, ' ', '') AS street_address_no_spaces
2  FROM locations;
3
4
```

Results | Explain | Describe | Saved SQL | History

STREET_ADDRESS_NO_SPACES
460BloorSt.W.
MagdalenCentre,TheOxfordSciencePark
2014JabberwockyRd
2011InteriorsBlvd
2004CharadeRd
Av.RioBranco

6 rows returned in 0.00 seconds Download