Angela Zhang

Programming 14-20

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

Vocabulary:

Unique Constraint	Every value in a column or set of columns (a composite key) must be unique
Not NULL constraint	Every value in a column or set of columns (a composite key) must be unique
Primary Key Constraint	Constraint ensures that the column contains no null values and uniquely identifies each row of the table
CHeck constraint	Specifies a condition for a column that must be true for each row of data
Foreign key	Identifies that table and column in the parent table
Uniqie constraint	An integrity constraint that requires every value in a column or set of columns be unique
Foreign Key Constraint	Designates a column (child table) that establishes a relationship between a primary key in the same table and a different table (parent table)
Out-of-line Constraint	References one or more columns and is defined separately from the definitions of the columns in the table
Constraint	Database rule
Column-level constraint	Database rule that references a single column

1. Define the term "constraint" as it relates to data integrity

    a. Constraints are any rules or conditions that need to be followed in a table to ensure data integrity

2. State when it is possible to define a constraint at the column level, and when it is possible at the table level
    a. Column-level constraint limitations apply to a single column. They are defined right after the column name.
    b. Table-Level constraints limitations are applied after defining a column and has the ability to enforce complex or many columns.
3. State why it is important to give meaningful names to constraints
    a. to ensure that there is a clear understanding of the purpose of interoperability
4. State which data integrity rules are enforced by NOT NULL and UNIQUE constraints
    a. VARCHAR2: For variable-length character data
    b. CHAR: fixed-length character data
    c. NUMBER
    d. DATE
5. Write a CREATE TABLE statement which includes NOT NULL and UNIQUE constraints at the table and column levels
    a. CREATE TABLE employees (employee_id NUMBER(6) CONSTRAINT pk_employee_id PRIMARY KEY, first_name VARCHAR2(50) NOT NULL, -This column cannot be NULL last_name VARCHAR2(50), -- This column can be NULL by default hire_date DATE, salary NUMBER(10, 2) DEFAULT 0 NOT NULL);
6. Explain how constraints are created at the time of table creation
    a. CREATE TABLE locations (location_id NUMBER(6) CONSTRAINT pk_location_id PRIMARY KEY, -- Primary key on location_id location_name VARCHAR2(100) NOT NULL, -- Location name cannot be NULL address VARCHAR2(255), -- Address is nullable by default city VARCHAR2(50) NOT NULL, -- City is mandatory postal_code VARCHAR2(20), -- Postal code is nullable country VARCHAR2(50) NOT NULL, -- Country is mandatory latitude

NUMBER(9, 6), -- Latitude (nullable) longitude NUMBER(9, 6), -- Longitude (nullable) created_date DATE DEFAULT SYSDATE NOT NULL -- Date when the location was created (defaults to current date);

7. Execute the CREATE TABLE statement in Oracle Application Express
    a. CREATE TABLE locations (location_id NUMBER(6) CONSTRAINT pk_location_id PRIMARY KEY, location_name VARCHAR2(100) NOT NULL, address VARCHAR2(255), city VARCHAR2(50) NOT NULL, postal_code VARCHAR2(20), country VARCHAR2(50) NOT NULL, latitude NUMBER(9, 6), longitude NUMBER(9, 6), created_date DATE DEFAULT SYSDATE NOT NULL);
8. Execute a DESCRIBE command to view the Table Summary information.
    a. DESCRIBE locations;
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
    a. CREATE TABLE locations (location_id NUMBER(6) CONSTRAINT pk_location_id PRIMARY KEY, location_name VARCHAR2(100) NOT NULL, VARCHAR2(255), city VARCHAR2(50) NOT NULL, postal_code VARCHAR2(20), country VARCHAR2(50) NOT NULL, latitude NUMBER(9, 6), longitude NUMBER(9, 6), created_date DATE DEFAULT SYSDATE NOT NULL, CONSTRAINT uq_location_name UNIQUE (location_name), -- Ensures unique location names CONSTRAINT uq_postal_code UNIQUE (postal_code));

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

ON DELETE CASCADE	Allows a foreign key row that is referenced to a primary key row to be deleted
Check Constraint	Explicitly defines a condition that must be met
Primary Key	A column or set of columns that uniquely identifies each row in a table
Not NULL constraint	Constraint ensures that the column contains no null values
ON DELETE SET NULL	Allows a child row to remain in a table with null values when a parent record has been deleted
Foreign Key Constraint	Establishes a relationship between the foreign key column and a primary key or unique key in the same table or a different table

Try It / Solve It

1. What is the purpose of a
    a. PRIMARY KEY - Provides a unique identifier
    b. FOREIGN KEY - establishes a relationship between two tables
    c. CHECK CONSTRAINT - ensures values meet specific conditions
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.
3. Create the animals table. Write the syntax you will use to create the table.
    a. animal_id NUMBER(6)
    b. name VARCHAR2(25)
    c. license_tag_number NUMBER(10)
    d. admit_date DATE

   e. adoption_id NUMBER(5),
   f. vaccination_date DATE
4. Enter one row into the table. Execute a SELECT * statement to verify your input. Refer to the graphic below for input.
   a. INSERT INTO animals (animal_id, name, license_tag_number, admit_date, adoption_id, vaccination_date)
   b. VALUES (101, 'Buddy', 1234567890, TO_DATE('2024-11-01', 'YYYY-MM-DD'), 201, TO_DATE('2024-11-15', 'YYYY-MM-DD'));
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.
   a. 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) REFERENCES adoptions(adoption_id), vaccination_date DATE NOT NULL);
6. What is the effect of setting the foreign key in the ANIMAL table as
   a. ON DELETE CASCADEm - parent tables is deleted and all the rows from animals table are also deleted
   b. ON DELETE SET NULL - It means that the parent table is deleted but the animal table will still have key to parent table is now set to null
7. What are the restrictions on defining a CHECK constraint?
   a. Conditions for check constraints must have a true or false value

14-3: Managing Constraints

Vocabulary

Disable Constraint	To deactivate an integrity constraint
CASCADE constraint disable	Disables dependent integrity constraints
Alter Table	To add, modify, or drop columns from a table
Enable constraint	To activate an integrity constraint currently disabled
Drop Constraint	Removes a constraint from a table
Drop Column	Allows user to delete a column from a table
On delete/on update clause	Defines the actions the database server takes when a user attempts to delete or update a key to which existing foreign keys point

1. What are four functions that an ALTER statement can perform on constraints?
   a. Constraint
   b. Drop
   c. Modify
   d. Rename
2. Since the tables are copies of the original tables, the integrity rules are not passed onto the newtables; only the column datatype definitions remain. You will need to add a PRIMARY KEYconstraint 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?
   a. ALTER TABLE copy_d_clients

ADD CONSTRAINT copy_d_clients_pk PRIMARY KEY (client_number);

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?
    a. ALTER TABLE copy_d_events

       ADD CONSTRAINT copy_d_events_fk FOREIGN KEY (client_number)

       REFERENCES copy_d_clients (client_number);
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 PRIMARY KEY
    b. The constraint name for the foreign key in the copy_d_events table is FOREIGN KEY
5. Drop the PRIMARY KEY constraint on the copy_d_clients table. Explain your results
    a. ALTER TABLE copy_d_clients

       DROP CONSTRAINT copy_d_clients_pk;
6. 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
    a. ALTER TABLE copy_d_clients

       DISABLE CONSTRAINT copy_d_clients_pk;
7. Repeat question 6: Insert the new values in the copy_d_events table Explain your results
    a. INSERT INTO copy_d_events (client_number, event_date, event_details)

       VALUES (12345, TO_DATE('2024-11-22', 'YYYY-MM-DD'), 'Event 2');
8. Enable the primary-key constraint in the copy_d_clients table
    a. ALTER TABLE copy_d_clients

       ENABLE CONSTRAINT copy_d_clients_pk;

9. If you wanted to enable the foreign-key column and reestablish the referential integrity between these two tables, what must be done?
    a. ALTER TABLE copy_d_events
       ENABLE CONSTRAINT copy_d_events_fk;
10. Why might you want to disable and then re-enable a constraint?
    a. to maintain referential integrity
    b. improve performance of the database
    c. evolve schema changes in infrastructure
11. Query the data dictionary for some of the constraints that you have created. How does the data dictionary identify each constraint type

    SELECT constraint_name, constraint_type, table_name

FROM user_constraints

WHERE table_name IN ('COPY_D_CLIENTS', 'COPY_D_EVENTS');

15-1: Creating Views

Objectives

- List three uses for views from the standpoint of a database administrator
- Explain, from a business perspective, why it is important to be able to create and use logical subsets of data derived from one or more tables
- Create a view with and without column aliases in the subquery using a single base table
- Create a complex view that contains group functions to display values from two tables
- Retrieve data from a view

Vocabulary

VIew	A subset of data one or more tables that is generated from a query and stored as a virtual table
View Name	Name of view View Name

Create force view	Creates a view regardless of whether or not the base tables exist
SImple view	Derives data from a table , no functions or groups, performs DML operations through the view
Create noforce view	Create the view only if the base table exists
Create view	Statement used to create a new view
Column alias	Specifies a name for each expression selected by the view's query
View query	A complete SELECT statement
Complex view	Derives data from more than one table , contains functions or groups of data, and does not always allow DML operations through the view
Create or replace view	Re-create the view if it already exists

Try It / Solve It

1. What are three uses for a view from a DBA's perspective?
   a. Data Security
   b. Simplification
   c. data abstraction
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
   a. CREATE OR REPLACE VIEW view_d_songs AS SELECT id, title AS "Song Title", artist
   FROM djs_on_demand
   WHERE type_code = 'New Age';

3. SELECT *FROM view_d_songs What was returned
    a. ID
    b. Song
    c. artist
4. REPLACE view_d_songs. Add type_code to the column list. Use aliases for all columns
    a. CREATE OR REPLACE VIEW view_d_songs AS

       SELECT id AS "Song ID", title AS "Song Title", artist AS "Artist Name", type_code AS "Type Code" FROM djs_on_demand

       WHERE type_code = 'New Age';
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.
    a. CREATE OR REPLACE VIEW jason_event_view AS

       SELECT event_name AS "Event Name", event_date AS "Event Date", theme_description AS "Theme" FROM events;
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
    a. CREATE OR REPLACE VIEW dept_salary_stats AS

       SELECT department_id AS "Department ID", MIN(salary) AS "Minimum Salary", MAX(salary) AS "Maximum Salary", AVG(salary) AS "Average Salary"

       FROM employees

       GROUP BY department_id;

15-2: DML Operations and Views

Objectives

- Write and execute a query that performs DML operations on a simple view
- Name the conditions that restrict modifying a view using DML operations
- Write and execute a query using the WITH CHECK OPTION clause
- Explain the use of WITH CHECK OPTION as it applies to integrity constraints and data validation
- Apply the WITH READ ONLY option to a view to restrict DML operations

Vocabulary

Identify the vocabulary word for each definition below.

rownum	pseudocolumn which assigns a sequential value starting with 1 to each of the rows returned from the subquery
With check option	Specifies that insert and update performed through the view can't create rows which the view cannot select
With read only	Ensures that no DML operations can be performed on this view

Try It / Solve It

1. Query the data dictionary USER_UPDATABLE_COLUMNS to make sure the columns in the base tables will allow updating, INSERT, or DELETE. Use a SELECT statement. All table names in the data dictionary are stored in uppercase.
    a. SELECT TABLE_NAME, COLUMN_NAME, UPDATABLE
       FROM USER_UPDATABLE_COLUMNS
       WHERE UPDATABLE = '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.
    a. CREATE OR REPLACE VIEW view_copy_d_songs AS SELECT * FROM copy_d_songs;

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. See the graphic.
    a. INSERT INTO view_copy_d_songs (column1, column2, column3, ...)
       VALUES (value1, value2, value3, ...);
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.
    a. CREATE OR REPLACE VIEW read_copy_d_cds AS
       SELECT *
       FROM COPY_D_CDS
       WHERE year = 2000
5. Using the read_copy_d_cds view, execute a DELETE FROM read_copy_d_cds WHERE cd_number = 90;
    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.
    a. 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;
7. Use the read_copy_d_cds view to delete any CD of year 2000 from the underlying copy_d_cds.
    a. DELETE FROM read_copy_d_cds WHERE year = 2000;
8. Use the read_copy_d_cds view to delete cd_number 90 from the underlying copy_d_cds table.
    a. DELETE FROM read_copy_d_cds WHERE cd_number = 90;
9. Use the read_copy_d_cds view to delete year 2001 records.
    a. DELETE FROM read_copy_d_cds WHERE year = 2001;

15-3: Managing Views

Objectives

- Create and execute a query that removes a view
- Create and execute a query using an inline view
- Create and execute a top-n-analysis query

Vocabulary

USER	Asks for the N largest or smallest values in a column
TRANSACTION	Removes a view
EXPLICIT	Subquery with an alias that can be used within a SQL statement

Try It / Solve It

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.
   a. CREATE VIEW view_copy_d_songs AS

      SELECT title, artist

      FROM copy_d_songs;

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

      SELECT * FROM view_copy_d_songs;
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.
   a. SELECT last_name, salary, RANK() OVER (ORDER BY salary DESC) AS rank

      FROM employees

WHERE RANK() OVER (ORDER BY salary DESC) <= 3;

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.
   a. SELECT last_name, salary, department_id, max_salary

      FROM employees

      JOIN (SELECT department_id, MAX(salary) AS max_salary

      FROM employees

      GROUP BY department_id)

      ON department_id = department_id;

5. Create a query that will return the staff members of Global Fast Foods ranked by salary from lowest to highest.
   a. SELECT staff_name, salary, RANK() OVER (ORDER BY salary ASC) AS rank

      FROM global_fast_foods_staff;

16-1: Working with Sequences

__**Objectives**__

- List at least three useful characteristics of a sequence
- Write and execute a SQL statement that creates a sequence
- Query the data dictionary using USER_SEQUENCES to confirm a sequence definition
- Apply the rules for using NEXTVAL to generate sequential numbers for use in a table
- List the advantages of caching sequence values
- Name three reasons why gaps can occur in a sequence

__**Vocabulary**__

Identify the vocabulary word for each definition below.

1. Command that automatically generates sequential numbers: AUTO_INCREMENT
2. Generates a numeric value: NEXTVAL

3. Returns the next available sequence value: NEXTVAL
4. Specifies the interval between sequence numbers: INCREMENT BY
5. Specifies a maximum value of 10^27 for an ascending sequence and -1 for a descending sequence (default): MAXVALUE
6. Returns the current sequence value: CURRVAL
7. Specifies the minimum sequence value: MINVALUE
8. Specifies whether the sequence continues to generate values after reaching its maximum or minimum values: CYCLE | NO CYCLE
9. Specifies a minimum value of 1 for an ascending sequence and – (10^26) for a descending sequence (default): MINVALUE
10. Specifies a maximum or default value the sequence can generate: MAXVALUE
11. Specifies the first sequence number to be generated: START WITH
12. Specifies how many values the server pre-allocates and keeps in memory: CACHE

Try It / Solve It

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.

   ```
   CREATE TABLE seq_d_songs AS
   SELECT *
   FROM d_songs;
   ```

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.

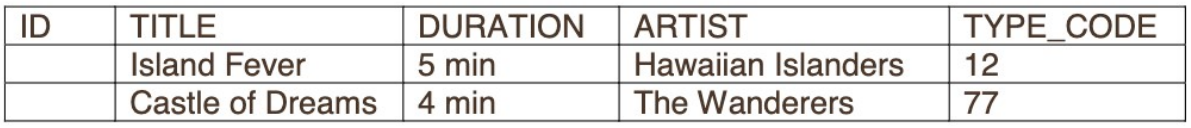
   ```
   CREATE SEQUENCE seq_d_songs_seq
   START WITH 100
   INCREMENT BY 2
   ```

```
MAXVALUE 1000
NOCACHE
NOCYCLE;
```

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

```
SELECT SEQUENCE_NAME,
MIN_VALUE,
MAX_VALUE,
INCREMENT_BY,
CYCLE_FLAG,
CACHE_SIZE,
LAST_NUMBER
FROM USER_SEQUENCES
WHERE SEQUENCE_NAME = 'SEQ_D_SONGS_SEQ';
```

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.



ID	TITLE	DURATION	ARTIST	TYPE_CODE
	Island Fever	5 min	Hawaiian Islanders	12
	Castle of Dreams	4 min	The Wanderers	77

```
INSERT INTO seq_d_songs (id, title, artist, genre)
VALUES (seq_d_songs_seq.NEXTVAL, 'Island Fever");

INSERT INTO seq_d_songs (id, title, artist, genre)
VALUES (seq_d_songs_seq.NEXTVAL, Castle of Dreams');
```

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?

Automatic numbering, it works well with multiple users, there are customizable settings

7. What are the advantages of caching sequence values?

Improved performance, reduced contention, less overhead 8.

Name three reasons why gaps may occur in a sequence?

Rollback of transactions, cache and pre-allocated values, manual sequence adjustments

Extension Exercise

1. Create a table called "students". You can decide which columns belong in that table and what datatypes these columns require. (The students may create a table with different columns; however, the important piece that must be there is the student_id column with a numeric datatype. This column length must allow the sequence to fit, e.g. a column length of 4 with a sequence that starts with 1 and goes to 10000000 will not work after student
#9999 is entered.)

   CREATE TABLE students (student_id NUMBER(8), first_name VARCHAR2(50), last_name VARCHAR2(50), date_of_birth DATE, email VARCHAR2(100), enrollment_date DATE);

2. Create a sequence called student_id_seq so that you can assign unique student_id numbers for all students that you add to your table.

   ```
   CREATE SEQUENCE student_id_seq
   START WITH 1
   INCREMENT BY 1
   MAXVALUE 99999999
   NOCACHE
   NOCYCLE;
   ```

3. Now write the code to add students to your STUDENTS table, using your sequence "database object."

   INSERT INTO students (student_id, first_name, last_name, date_of_birth, email, enrollment_date)

   VALUES (student_id_seq.NEXTVAL, 'John', 'Doe', TO_DATE('2000-05-15', 'YYYY-MM-DD'), 'john.doe@example.com', SYSDATE);

INSERT INTO students (student_id, first_name, last_name, date_of_birth, email, enrollment_date)

VALUES (student_id_seq.NEXTVAL, 'Jane', 'Smith', TO_DATE('1999-08-22', 'YYYY-MM-DD'), 'jane.smith@example.com', SYSDATE);

16-2: Indexes and Synonyms

Objectives

- Define an index and its use as a schema object
- Name the conditions that cause an index to be created automatically
- Create and execute a CREATE INDEX and DROP INDEX statement
- Construct and execute a function-based index
- Construct a private and public synonym

Vocabulary

Identify the vocabulary word for each definition below.

- Confirms the existence of indexes from the USER_INDEXES data dictionary view SELECT
- Schema object that speeds up retrieval of rows INDEX
- To refer to a table by another name to simplify access ALIAS
- An index that you create on multiple columns in a table COMPOSITE INDEX
- The Oracle Server automatically creates this index when you define a column in a table to have a PRIMARY KEY or a UNIQUE KEY constraint UNIQUE INDEX
- Stores the indexed values and uses the index based on a SELECT statement to retrieve the data INDEX TABLE
- Removes an index DROP INDEX
- Gives alternative names to objects ALIAS

Try It / Solve It

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

   An index is a tool in a database that helps find data faster.

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

   A ROWID is a unique identifier for each row in a database table.

3. When will an index be created automatically?

   An index is created automatically when you define a column with a PRIMARY KEY or UNIQUE constraint.

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.

   ```
   CREATE INDEX idx_cd_number
   ON D_TRACK_LISTINGS (cd_number);
   ```

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.

   ```
   SELECT i.index_name, uniqueness, column_name
   FROM user_indexes
   JOIN user_ind_columns
   ON index_name = index_name
   WHERE table_name = 'D_SONGS';
   ```

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.

   ```
   SELECT index_name, table_name, uniqueness
   FROM user_indexes
   WHERE table_name = 'D_EVENTS';
   ```

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

   ```
   CREATE SYNONYM dj_tracks FOR d_track_listings;
   ```

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.

```
SELECT *
FROM D_PARTNERS
WHERE UPPER(last_name) = 'SMITH';
```

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

```
CREATE SYNONYM dj_track_listings FOR D_TRACK_LISTINGS;
```

10. Drop the synonym that you created in question 9. DROP SYNONYM dj_track_listings;

17-1: Controlling User Access

Objectives

- Compare the difference between object privileges and system privileges
- Construct the two commands required to enable a user to have access to a database
- Construct and execute a GRANT... ON ...TO statement to assign privileges to objects in a user's schema to other users and/or PUBLIC
- Query the data dictionary to confirm privileges granted

Try It / Solve It

1. What are system privileges concerned with?

   System privileges are concerned with actions that a user can perform on the database system itself, such as creating tables, altering schemas, or managing users.

2. What are object privileges concerned with?

   Object privileges are concerned with actions that a user can perform on specific objects (like tables, views, or procedures) within the database, such as SELECT, INSERT, UPDATE, or DELETE.

3. What is another name for object security?

   Object security is also known as 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.

   SELECT *

   FROM user_tab_privs;

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

   GRANT CREATE TABLE TO user_name;

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 user_name;

10. What syntax would you use to grant another user access to your copy_employees table?
    GRANT SELECT, INSERT, UPDATE, DELETE ON copy_employees TO user_name;

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

    SELECT *

    FROM user_tab_privs_columns;

17-2: Creating and Revoking Object Privileges

Objectives

- Explain what a ROLE is and what its advantages are
- Construct a statement to create a ROLE and GRANT privileges to it
- Construct a GRANT ON TO WITH GRANT OPTION statement to assign privileges to objects in a user's schema to other users and/or PUBLIC
- Construct and execute a statement to REVOKE object privileges from other users and/or from PUBLIC
- Distinguish between privileges and roles
- Explain the purposes of a database link

Try It / Solve It 1.

What is a role?

A role is a collection of privileges that can be granted to users or other roles. Roles are used to simplify the management of permissions by grouping together multiple privileges.

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

   Simplifying user and privilege management by grouping related privileges.Efficiently managing access control, especially when there are many users with similar privileges.

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 your_table TO user_name 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?

   You should use roles to make your job easier. Roles allow you to group the system privileges and assign them to users without needing to grant privileges individually.

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;

a. 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;

b. Grant the manager role to user scott

   GRANT manager TO scott;

c. 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?

   Allows a user to access objects in another database.

17-3: Regular Expressions

Objectives

- Describe regular expressions
- Use regular expressions to search, match, and replace strings in SQL statements
- Construct and execute regular expressions and check constraints

Try It / Solve It

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).

   SELECT *

   FROM employees

   WHERE REGEXP_LIKE(first_name, '^S[t|h]', 'i');

2. Investigate the LOCATIONS table.

a. Describe the table.

   DESCRIBE LOCATIONS;

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

```
SELECT *
FROM LOCATIONS;
```

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

```
SELECT REGEXP_REPLACE(street_address, ' ', '') AS street_address_no_spaces
FROM LOCATIONS;
```

18-1: Database Transactions

Objectives

- Define the COMMIT, ROLLBACK, and SAVEPOINT statements as they relate to data transactions
- List three advantages of COMMIT, ROLLBACK, and SAVEPOINT statements
- Explain why it is important, from a business perspective, to be able to control the flow of transaction processing

Vocabulary

Identify the vocabulary word for each definition below

- Ends the current transaction making all pending data changes permanent Commit
- Enables the user to discard changes made to the database Rollback
- Creates a marker in a transaction, which divides the transaction into smaller pieces guarantees a consistent view of the data by all users at all times Savepoint
- Mechanisms that prevent destructive interaction between transactions accessing the same resource that can be granted to the user a collection of DML statements that form a logical unit of work Locking/Transaction

Try It / Solve It

1. Define the COMMIT, ROLLBACK, and SAVEPOINT statements as they relate to data

Transactions.

Commit: end current transactions and make all changes permanent

Rollback: undo all changes made during current transaction

Savepoint: create name marker within transaction to allow partial rollback

2. What data will be committed after the following statements are issued?

- INSERT INTO R values (5, 6);
- SAVEPOINT my_savepoint_1;
- INSERT INTO R values (7, 8);
- SAVEPOINT my_savepoint_2;
- INSERT INTO R values (9, 10);
- ROLLBACK TO my_savepoint_1;
- INSERT INTO R values (11, 12);
- COMMIT;

(5, 6)

(11, 12)

3. Construct a SQL statement for the DJs on Demand D_SONGS table that deletes the song “All These Years,” inserts a new Country song called ‘Happy Birthday Sunshine’ by “The Sunsets” with a duration of 4 min and an ID = 60. Make sure that all data can be recovered before any changes to the table are made.

```
SAVEPOINT before_changes;
DELETE FROM D_SONGS WHERE song_name = 'All These Years';
INSERT INTO D_SONGS (song_id, song_name, artist, genre, duration)
VALUES (60, 'Happy Birthday Sunshine', 'The Sunsets', 'Country', 4);
ROLLBACK TO before_changes;
COMMIT;
```

4. Write an SQL statement that will issue an automatic commit.

```
COMMIT;
```

5. Give two examples of businesses other than banks that rely on transaction control processes. Describe why each business needs transaction processing control.

Amazon

Airline

19-1: Testing

Objectives

- Develop and apply a strategy for testing that a database functions as designed.

Try It / Solve It

1. Design and carry out tests to check the following:

   a. The business rule that requires that employees have a job_id

   SELECT employee_id, first_name, last_name

   FROM employees

   WHERE job_id IS NULL;

   b. The business rule that requires that the end date of an employment is after a start date in the job history table.

   SELECT employee_id, job_id, start_date, end_date

   FROM job_history

   WHERE end_date < start_date;

   c. The business rule states that departments can be closed down with employees in that department (resulting in the department_id becoming unknown).

   SELECT employee_id, first_name, last_name, department_id

   FROM employees

   WHERE department_id IS NULL;

   d. The minimum salary of an employee is 1000.

   SELECT employee_id, first_name, last_name, salary

   FROM employees

   WHERE salary < 1000;

2. If one of the above tests fails, write out the SQL statement(s) that would be needed to correct the test. With the permission of your teacher, implement the change and then rerun the test with the same input and confirm that it works.
   a. UPDATE employees
      SET job_id = 'GEN'
      WHERE job_id IS NULL;
   b. UPDATE job_history
      SET end_date = start_date + INTERVAL '1 DAY'
      WHERE end_date < start_date;
   c. UPDATE employees
      SET department_id = 999
      WHERE department_id IS NULL;
   d. UPDATE employees
      SET salary = 1000
      WHERE salary < 1000;

19-3: Final Exam Review

Objectives

- Review the key points about case and character manipulation
- Review number, date, conversion, and general functions
- Review conditional expressions
- Review Cartesian product and join operations
- Review non-equijoins, outer joins, self joins, cross joins, natural joins, and join clauses
- Review group functions, group by syntax, and having clauses
- Review single-row and multiple row subqueries
- Review pair-wise and non-pair-wise subqueries
- Review correlated subqueries
- Review DML statements insert, update, delete, merge, and multi-table inserts
- Review DDL statements CREATE, ALTER, RENAME, TRUNCATE, FLASHBACK

TABLE, DROP, and FLASHBACK QUERY

- Review DCL statements CREATE and REVOKE object privileges

Try It / Solve It

1. The business manager of Global Fast Foods needs to update the customer list. She wants to find any zip/postal code that has fewer than 10 digits in order to identify those codes without the new format postfix 87392-8723. Create a query to identify those customers.

```
SELECT customer_id, customer_name, zip_code
FROM customers
WHERE LENGTH(zip_code) < 10;
```

2. In the following list of functions, mark with N those that can be used with numbers, mark with C those that can operate on character data, and mark with D those that can be used with dates.

   a. LPAD C
   b. ROUND N
   c. TRUNC N
   d. LENGTH C
   e. LAST_DAY N
   f. INSTR C
   g. CONCAT C

3. You need to display the auth_expense_amt for each DJs on Demand partner. For those partners who do not have an expense account, the output should display "Not Approved."

```
SELECT partner_id, NVL(auth_expense_amt, 'Not Approved') AS expense_status
FROM partners;
```

4. Jason and Jamie tried to run a report displaying the Global Fast Foods staff members who do not have an overtime rate. They wrote the following SQL query and received a report error:

ORA-01722: invalid number.

```
SELECT first_name, last_name, NVL(overtime_rate, 'no overtime') As "Payrate"
FROM f_staffs;
```

What is wrong with their query and how can it be fixed?

SELECT first_name, last_name, NVL(overtime_rate, NULL) AS "Payrate"

FROM f_staffs;

5. The president of Global Fast Foods likes to send a birthday card to all employees. He has asked you to send a reminder to him on the month of the employee's birthday so the card can be sent with that month's paycheck. Prepare a query to produce the table shown.

FIRST_NAME LAST_NAME BIRTHDATE SEND CARD

Sue Doe 01-Jul-1980 July 2005

Bob Miller 19-Mar-1979 March 2005

Monique Tuttle 30-Mar-1969 March 2005

SELECT first_name, last_name, TO_CHAR(birthdate, 'DD-Mon-YYYY') AS birthdate,

TO_CHAR(ADD_MONTHS(birthdate, 0), 'Month YYYY') AS send_card

FROM employees

WHERE EXTRACT(MONTH FROM birthdate) = EXTRACT(MONTH FROM SYSDATE);

6. For each statement, mark T if the statement is true or F if the statement is false.

a. TO_CHAR is required to convert the date '03-Jun-2004' to June 3, 2004. True

b. TO_NUMBER will convert '23-Nov-2002' to use with ADD_MONTHS. False

c. TO_DATE will convert SYSDATE to today's date. False

d. TO_NUMBER('101', '$99999') will convert 101 to a number. False

e. TO_CHAR(salary, '$9999.99') will convert number to character format. True

f. TO_NUM(varchar2 column) will convert character data to a number. False

g. TO_CHAR(SYSDATE, 'Month fmdd, yyyy') will format the date. True

7. Employees in the Oracle database who have worked more than 10 years will be given one extra week of vacation. Create a report showing first name, last name, and years worked. Round the result to one decimal place.

SELECT first_name, last_name, ROUND(MONTHS_BETWEEN(SYSDATE, hire_date) / 12,1)

AS years_worked

FROM employees

WHERE MONTHS_BETWEEN(SYSDATE, hire_date) / 12 > 10;

8. The manager of DJs on Demand needs you to change the zip code for ID 105, New York venues. She wants to show the old zip code in one column and the new zip code in another column. The new zip code is the same as the old zip code except -2345 needs to be added to the end. The output should appear as shown in the table.

OLD ZIP NEW ZIP

11220 11220-2345

SELECT zip_code AS "OLD ZIP", CONCAT(zip_code, '-2345') AS "NEW ZIP"

FROM venues

WHERE venue_id = 105;

9. Create a query using one SELECT statement that returns today's date. Assign an alias to each column.

   a. Rounded to the nearest year

   b. Rounded to the nearest month

   c. Truncated to the nearest year

   d. Truncated to the nearest month

SELECT SYSDATE AS "Today", ROUND(SYSDATE, 'YYYY') AS "Rounded to Nearest Year", ROUND(SYSDATE, 'MM') AS "Rounded to Nearest Month", TRUNC(SYSDATE, 'YYYY') AS "Truncated to Nearest Year", TRUNC(SYSDATE, 'MM') AS "Truncated to Nearest Month"

FROM dual;

10. You need to find out how many days it has been since the start of the Global Fast Foods promotional menus. Round the result to the nearest day.

SELECT ROUND(SYSDATE - promo_start_date) AS days_since_promo_start

FROM promotions

WHERE promo_id = 1; -- Assuming promo_id 1 is for the promotional menus

11. The Human Resources department (Oracle database) has decided that, for their purposes, the job title for all employees will be the first five letters of the job title followed by an asterisk. For example, the accounting manager will be changed to accou* Create a query to accomplish this request.

UPDATE employees

SET job_title = CONCAT(SUBSTR(job_title, 1, 5), '*');

12. What is the order of operations in question 11?

    1. SUBSTR
    2. CONCAT
    3. UPDATE

13. Write a query to return all the rows and columns from the employees table, but make the department_id a substitution variables, and then execute your query with two or three different department_id's to test it.

SELECT *

FROM employees

WHERE department_id = &department_id;

2. Create two different queries that produce the cd number and title as "94CARPE DIEM." Use the d_cds table in the DJs on Demand database.

    a. SELECT CONCAT(cd_number, title) AS "94CARPE DIEM"

       FROM d_cds

       WHERE cd_number = 94 AND title = 'CARPE DIEM';

    b. SELECT cd_number || title AS "94CARPE DIEM"

       FROM d_cds

       WHERE cd_number = 94 AND title = 'CARPE DIEM';

3. Mark the following statements as True or False.

__F___a. LOWER converts numbers to lowercase.

__T___b. Use RPAD to move numbers to the right to place an * on the left.

__T___c. TRIM can be used to trim one or more characters from a string.

__T___d. LENGTH returns a number.

__F___e. SUBSTR is used to substitute one string for another.

__F___f. CONCAT is limited to using two parameters.

__T___g. TRUNC will return zero decimal places if a decimal value is omitted.

4. Create a query to show the cost of events for DJs on Demand in the format $0000.00

SELECT TO_CHAR(event_cost, '$0000.00') AS formatted_cost

FROM events;

5. For the f_staffs table in the Global Fast Foods database, change the ID of those staff members whose IDs are only single digits to two digits by adding an asterisk * to the front of the number. For example, change ID 9 to ID *9.

UPDATE f_staffs

SET staff_id = LPAD(staff_id, 2, '*')

WHERE LENGTH(staff_id) = 1;

6. As the database administrator, you have been asked to store historical employee records in the current database. The records have termination dates from 1990 to 1999. Write a query using DUAL to show how you could store 15-Dec-1995.

SELECT TO_DATE('15-Dec-1995', 'DD-Mon-YYYY') AS historical_date

FROM DUAL;

7. Using DUAL, format 19-Jun-2004 to appear as: 19th of june two thousand four SELECT TO_CHAR(TO_DATE('19-Jun-2004', 'DD-Mon-YYYY'), 'DDth "of" Month "two thousand" YYYY') AS formatted_date

FROM DUAL;

8. Create a query that will return only the last word from "Oracle Academy."

SELECT SUBSTR('Oracle Academy', INSTR('Oracle Academy', ' ', -1) + 1) AS last_word

FROM DUAL;

9. Lance and Arnie created the following SQL query but it did not return any results. What is the problem with this query?

SELECT loc_type

FROM d_venues

WHERE loc_type LIKE 'National Park'

AND ID = 200;

SELECT loc_type

FROM d_venues

WHERE loc_type = 'National Park'

AND ID = 200;

10. What type of function would you use in each case?

D = Date function

N = Number function

C = Conversion/Character functions

G = General function

CE = Conditional expression

___N__ a. To convert varchar2 to number data

___D__ b. To format a date to other than the default format

___D__ c. To convert a date such as June 19, 2000 to default format

___N__ d. To format a number to appear as currency

___CE__ e. To substitute a value in a table for null

___CE__ f. To do an IF-THEN-ELSE statement

___CE__ g. To find the first not null expression among a list of expressions

___C__ h. To replace a section of a string with another string

__D___ i. To format a 20th-century date

__C___ j. To present output all in uppercase

__C___ k. To find the numeric position of a character in a string

__D___ l. To find the last day of the month

1. A/An ___Cartesian Join____ is when the rows of the tables are combined with each other and produce new rows. The number of rows is equivalent to the product of the number of rows in each table.

2. A/An ___Self-Join__ is used when you need to query a table that has a relationship to itself. 3. A/An ____Outer Join____ preserves unmatched rows from one or both tables, returning the rows that are matched and unmatched from one or both tables.

4. In an outer join, a plus sign (+) is placed on the side of the join that is Missing_____information.

5. A _____Join Condition____is used when a column in one table does not correspond directly to a column in another table.

6. The join condition is always placed in the ____WHERE___ clause of the SELECT statement. 7. A/An ___Table Alias____ is used to preface the column name in order to clarify which table and column are participating in the join.

8. Table aliases are created in the ___FROM___ clause of the SELECT statement.

9. In a full outer join, a row that does not contain data will/will not appear in the result set if the row satisfies the join condition.

10. Table aliases cannot exceed __30__ characters in length.

11. Identify the Oracle syntax to signify an outer join___+___.

12. If a join condition is written: WHERE e.client_number = c.client_number, what kind of join would it be if we wanted all the information in the e table even if the c table has missing data?

__Left Outer Join____

13. Joins that are based on hierarchical relationships such as manager and employee are called __Recursive Joins__.

14. How many join conditions does it take to join three tables? _Two__

15. What does the term "proprietary syntax" mean? Vendor-Specific Syntax

20-1: Ensuring Quality Query Results– Advanced Technique

Objectives

- Create a query to produce specific data
- Modify a query to produce specified data

Try It / Solve It

2. Produce a report that lists the constraint name, type, column name, and column position of all the constraints on the JOB_HISTORY table, apart from the not null constraints.

```
SELECT constraint_name, constraint_type, column_name, position

FROM user_cons_columns

WHERE table_name = 'JOB_HISTORY'
```

AND constraint_type <> 'C' -- Exclude CHECK constraints, if you want to exclude all types

AND constraint_type <> 'N'; -- Exclude NOT NULL constraints

3. Create a primary key constraint on the emp table's employee_id column

ALTER TABLE emp

ADD CONSTRAINT pk_emp_employee_id PRIMARY KEY (employee_id);

4. Create a primary key on the dept table's department_id column

ALTER TABLE dept

ADD CONSTRAINT pk_dept_department_id PRIMARY KEY (department_id);

5. Add a foreign constraint between DEPT and EMP so that only valid departments can be entered in the EMP table. Make sure you can delete any row from the DEPT table, and that referenced rows in the EMP table are deleted.

ALTER TABLE emp

ADD CONSTRAINT fk_emp_dept

FOREIGN KEY (department_id) REFERENCES dept(department_id)

ON DELETE CASCADE;

6. Test the foreign key constraint you just created:

Count the number of rows in the EMP table. SELECT COUNT(*) FROM emp;

Remove department 10 from the dept table. DELETE FROM dept WHERE department_id = 10;

Now count emps again. There should be fewer employees. SELECT COUNT(*) FROM emp;

7. Produce a report that returns the last name, salary, department number, and average salary of all the departments where salary is greater than the average salary.

SELECT last_name, salary, department_id, (SELECT AVG(salary) FROM employees WHERE department_id = department_id) AS avg_salary

FROM employees

WHERE salary > (SELECT AVG(salary) FROM employees WHERE department_id = department_id);

8. Create a view named V2 that returns the highest salary, lowest salary, average salary and department name.

CREATE VIEW V2 AS

```
SELECT department_id, MAX(salary) AS highest_salary, MIN(salary) AS lowest_salary,
AVG(salary) AS avg_salary
FROM employees
GROUP BY department_id;
```

9. Create a view named Dept_Managers_view that returns a listing of department names long with the manager initial and surname for that department. Test the view by returning all the rows from it. Make sure no rows can be updated through the view. Try to run an UPDATE statement against the view.

```
CREATE VIEW Dept_Managers_view AS
SELECT department_name, first_name || ' ' || last_name AS manager_name
FROM departments d
JOIN employees ON manager_id = employee_id;
```

10. Create a sequence named ct_seq using all the default values.

```
CREATE SEQUENCE ct_seq;
```

11. Examine the following insert statement and fix the errors.

```
INSERT INTO emp (employee_id, first_name, last_name, email, phone_number, hire_date,
job_id, salary, commission_pct, manager_id, department_id)
VALUES (ct_seq.nextvalue, "Kaare", 'Hansen', 'KHANSEN', '44965 832123',
sysdate, 'SA_REP', $6500, null, 100, 20);
```

```
INSERT INTO emp (employee_id, first_name, last_name, email, phone_number, hire_date,
job_id, salary, commission_pct, manager_id, department_id)
VALUES (ct_seq.NEXTVAL, 'Kaare', 'Hansen', 'KHANSEN', '44965 832123', SYSDATE,
'SA_REP', 6500, NULL, 100, 20);
```

12. Write the SQL statement to list all the user tables which contains the name PRIV.

```
SELECT table_name
FROM user_tables
WHERE table_name LIKE '%PRIV%';
```

13. Give select access to public on the EMP table, and verify the grant by running this query.

```
SELECT *
FROM user_tab_privs
WHERE table_name = 'EMP';
GRANT SELECT ON emp TO PUBLIC;
```

14. Replace the ?? in the following query using regular expressions to return only the numbers from the following string: 'Oracle Academy9547d6905%&^ db apex'.

```
SELECT REGEXP_REPLACE('Oracle Academy9547d6905%&^ db apex',??,") regexpreplace
FROM DUAL;
SELECT REGEXP_REPLACE('Oracle Academy9547d6905%&^ db apex', '[^0-9]', ") AS
regexpreplace
FROM DUAL;
```

15. Amend the previous query using regular expressions to return the number of digits from the following string: 'Oracle Academy9547d6905%&^ db’

```
SELECT LENGTH(REGEXP_REPLACE('Oracle Academy9547d6905%&^ db apex','??',"))
regexpreplace
FROM DUAL;
SELECT LENGTH(REGEXP_REPLACE('Oracle Academy9547d6905%&^ db apex', '[^0-9]',
")) AS regexpreplace
FROM DUAL;
```

16. Amend the query again to return only the non-numeric characters.

```
SELECT REGEXP_REPLACE('Oracle Academy9547d6905%&^ db apex','??',") regexpreplace
FROM DUAL;
SELECT REGEXP_REPLACE('Oracle Academy9547d6905%&^ db apex', '[0-9]', ") AS
regexpreplace
FROM DUAL;
```

17. Using Oracle proprietary joins, construct a statement that returns all the employee_ids joined to all the department_names.

```
SELECT employee_id, department_name
FROM employees, departments;
```

18. Still using Oracle Joins, correct the previous statement so that it returns only the name of the department that the employee actually works in.

```
SELECT employee_id,department_name
FROM employees
JOIN departments ON department_id = department_id;
```

19. Still using Oracle Joins, construct a query that lists the employees last name, the department name, the salary, and the country name of all employees.

```
SELECT last_name, department_name, salary, country_id
FROM employees
JOIN departments ON department_id = department_id
JOIN locations l ON location_id = location_id;
```

20. Still using Oracle join syntax, alter the previous query so that it also includes the employee record of the employee with no department_id, ‘Grant’.

```
SELECT last_name
```