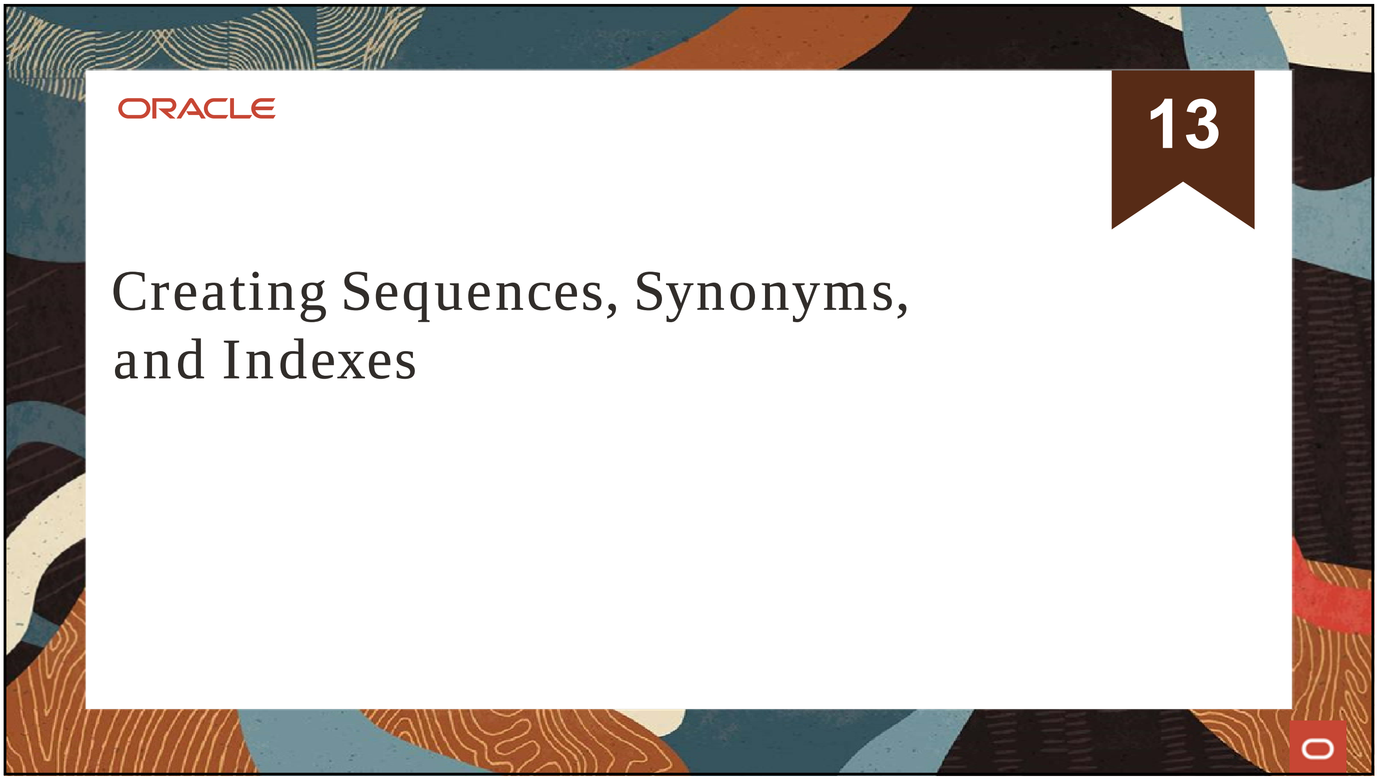
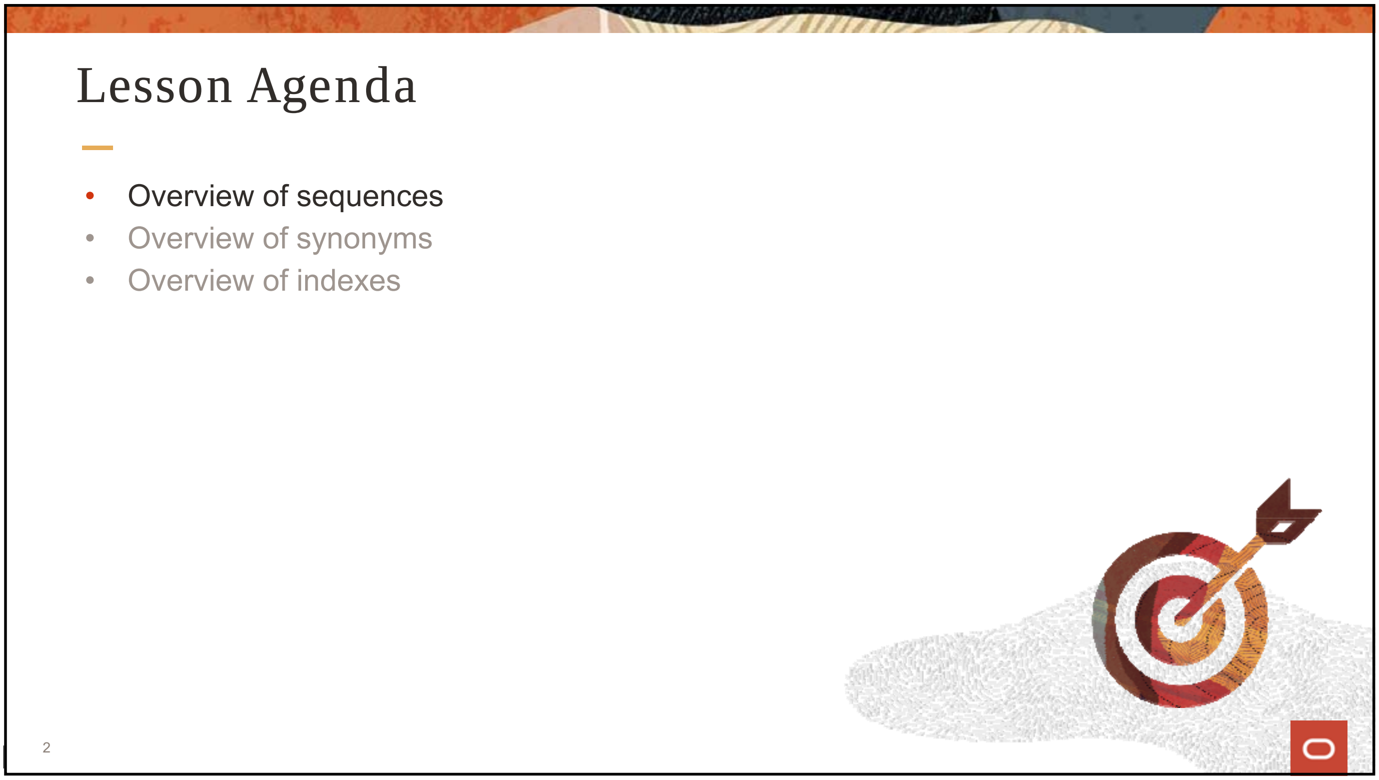


ORACLE

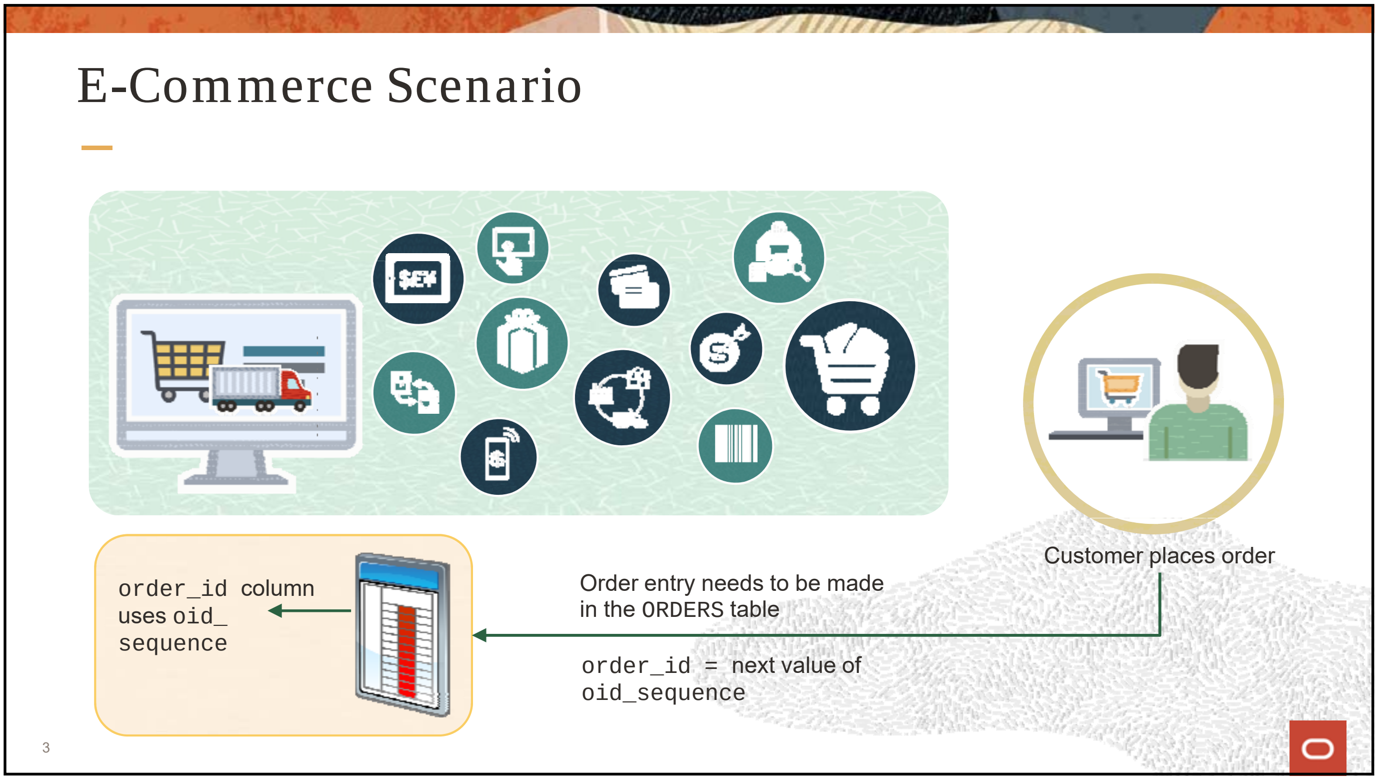
13

Creating Sequences, Synonyms, and Indexes



Lesson Agenda

- Overview of sequences
- Overview of synonyms
- Overview of indexes



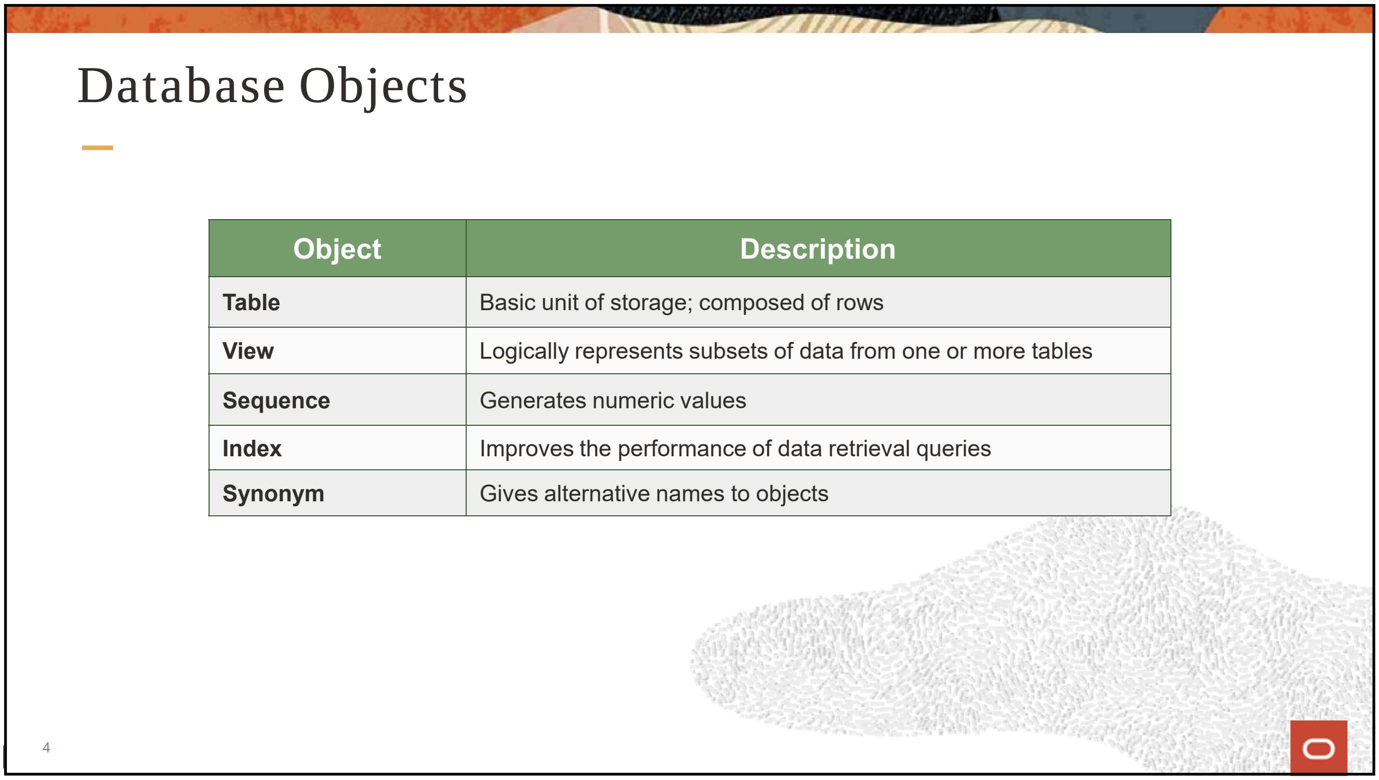
E-Commerce Scenario

Order entry needs to be made in the ORDERS table

order_id = next value of oid_sequence

order_id column uses oid_sequence

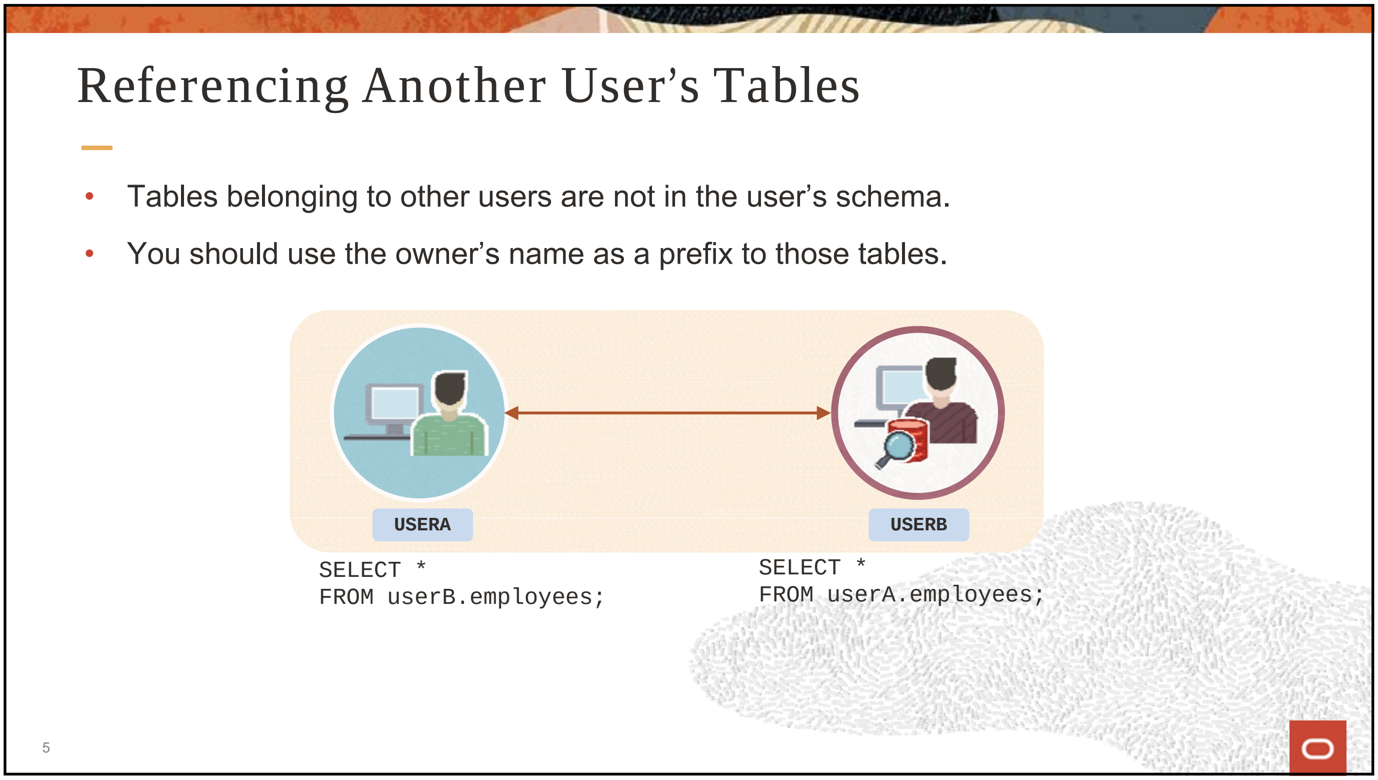
3



Database Objects

Object	Description
Table	Basic unit of storage; composed of rows
View	Logically represents subsets of data from one or more tables
Sequence	Generates numeric values
Index	Improves the performance of data retrieval queries
Synonym	Gives alternative names to objects

4



Referencing Another User's Tables

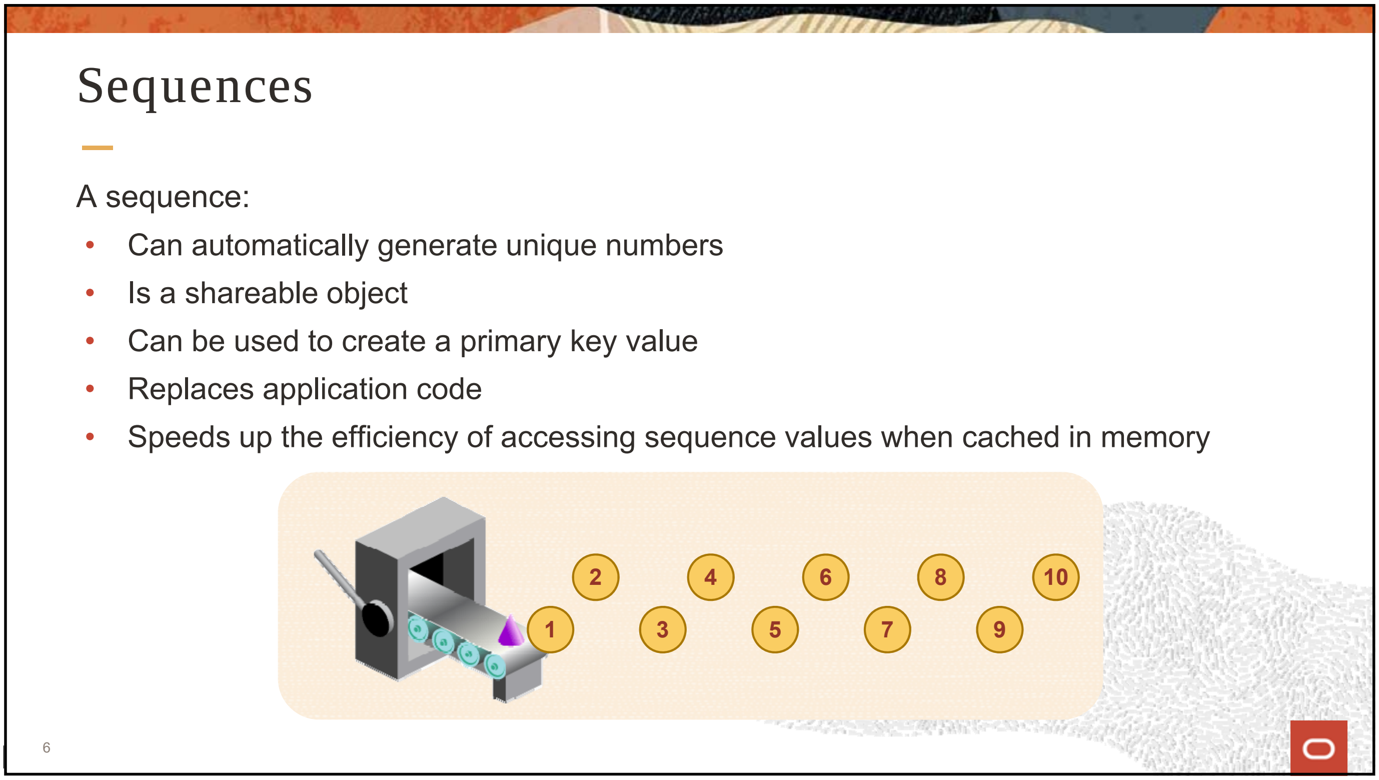
- Tables belonging to other users are not in the user's schema.
- You should use the owner's name as a prefix to those tables.

USERA
USERB

```
SELECT *
FROM userB.employees;
```

```
SELECT *
FROM userA.employees;
```

5



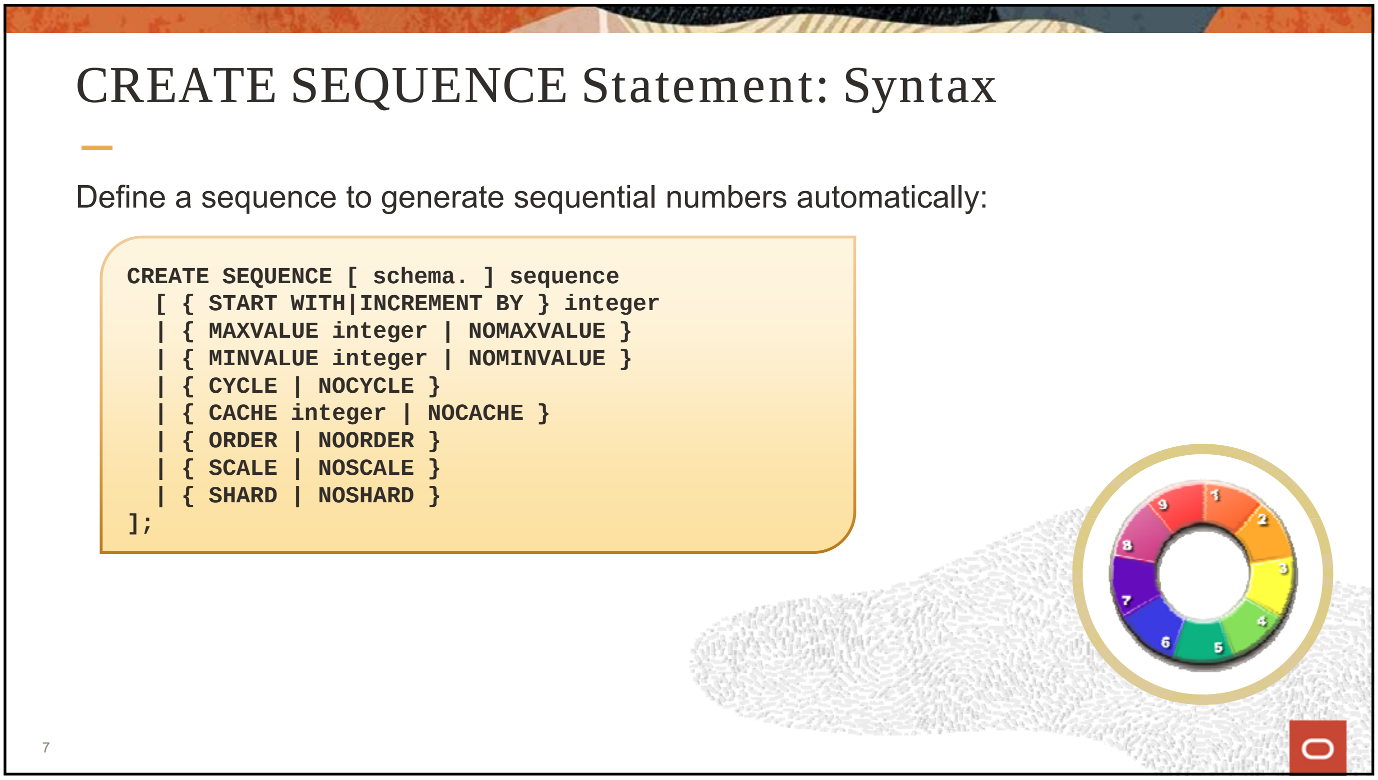
Sequences

A sequence:

- Can automatically generate unique numbers
- Is a shareable object
- Can be used to create a primary key value
- Replaces application code
- Speeds up the efficiency of accessing sequence values when cached in memory

2
4
6
8
10
1
3
5
7
9

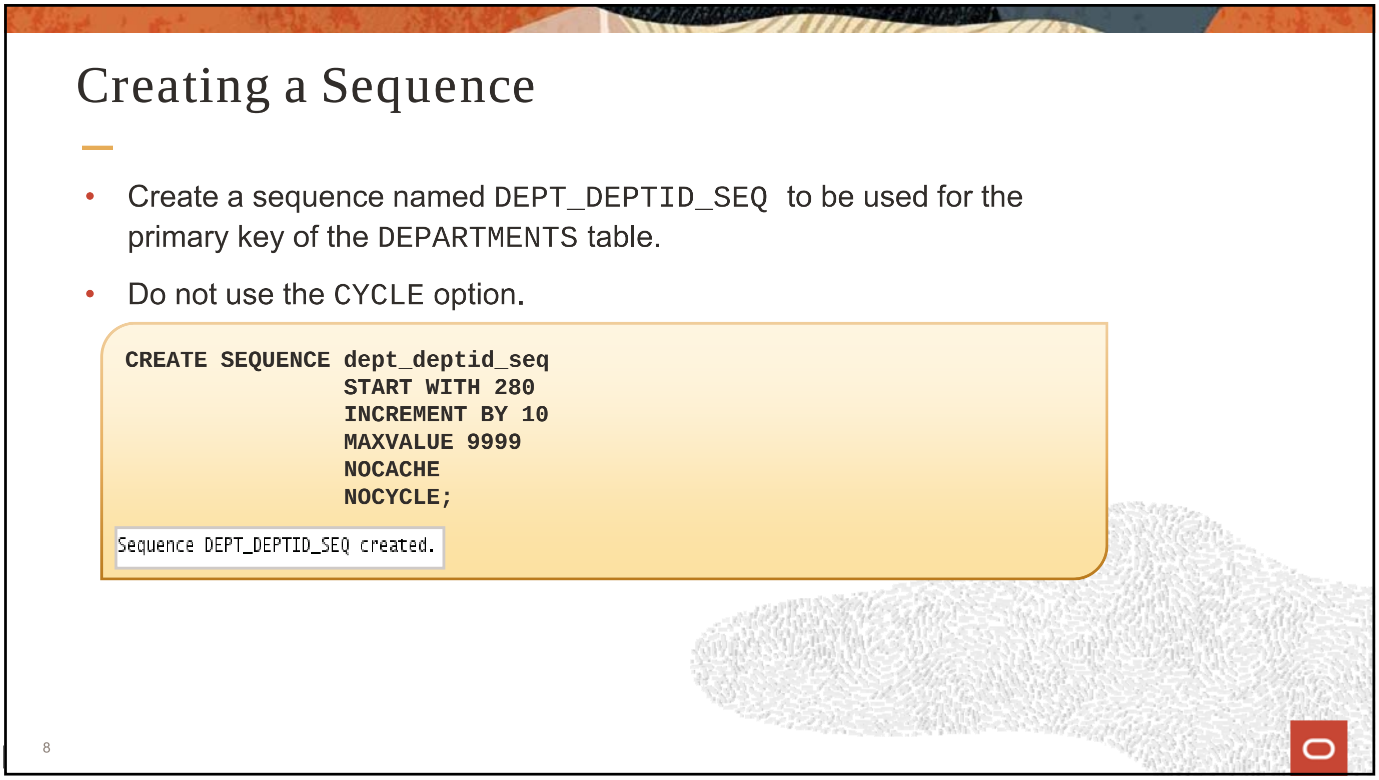
6



CREATE SEQUENCE Statement: Syntax

Define a sequence to generate sequential numbers automatically:

```
CREATE SEQUENCE [ schema. ] sequence
  [ { START WITH|INCREMENT BY } integer
  | { MAXVALUE integer | NOMAXVALUE }
  | { MINVALUE integer | NOMINVALUE }
  | { CYCLE | NOCYCLE }
  | { CACHE integer | NOCACHE }
  | { ORDER | NOORDER }
  | { SCALE | NOSCALE }
  | { SHARD | NOSHARD }
];
```



Creating a Sequence

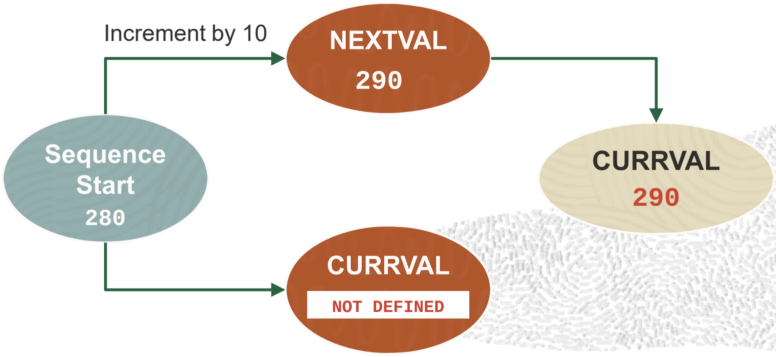
- Create a sequence named DEPT_DEPTID_SEQ to be used for the primary key of the DEPARTMENTS table.
- Do not use the CYCLE option.

```
CREATE SEQUENCE dept_deptid_seq
                START WITH 280
                INCREMENT BY 10
                MAXVALUE 9999
                NOCACHE
                NOCYCLE;
```

```
Sequence DEPT_DEPTID_SEQ created.
```

NEXTVAL and CURRVAL Pseudocolumns

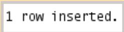
- NEXTVAL returns the next available sequence value. It returns a unique value every time it is referenced, even for different users.
- CURRVAL obtains the current sequence value.
- NEXTVAL must be issued for that sequence before CURRVAL can be referenced.



Using a Sequence

- Insert a new department named “Support” in location ID 2500:

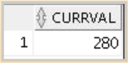
```
INSERT INTO depts(department_id,
            department_name, location_id)
VALUES      (dept_deptid_seq.NEXTVAL,
            'Support', 2500);
```



```
1 row inserted.
```

- View the current value for the DEPT_DEPTID_SEQ sequence:

```
SELECT dept_deptid_seq.CURRVAL
FROM dual;
```

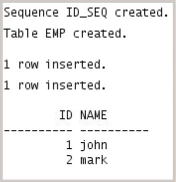


```
    CURRVAL
1       280
```

SQL Column Defaulting Using a Sequence

- You can use the SQL syntax `<sequence>.nextval, <sequence>.currval` as a SQL column defaulting expression for numeric columns, where <sequence> is an Oracle database sequence.
- The `DEFAULT` expression can include the sequence pseudocolumns `CURRVAL` and `NEXTVAL`, as long as the sequence exists and you have the privileges necessary to access it.

```
CREATE SEQUENCE ID_SEQ START WITH 1;
CREATE TABLE emp1 (ID NUMBER DEFAULT ID_SEQ.NEXTVAL NOT NULL,
                   name VARCHAR2(10));
INSERT INTO emp1 (name) VALUES ('john');
INSERT INTO emp1 (name) VALUES ('mark');
SELECT * FROM emp1;
```



```
Sequence ID_SEQ created.
Table EMP created.

1 row inserted.
1 row inserted.

        ID NAME
---------- ----------
         1 john
         2 mark
```

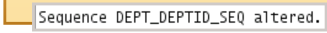
Gaps in sequence

- Gaps in sequence values can occur when:
  - A rollback occurs
  - The system crashes
  - A sequence is used in another table

Modifying a Sequence

Change the increment value, maximum value, minimum value, cycle option, or cache option:

```
ALTER SEQUENCE dept_deptid_seq
               INCREMENT BY 20
               MAXVALUE 999999
               NOCACHE
               NOCYCLE;
```

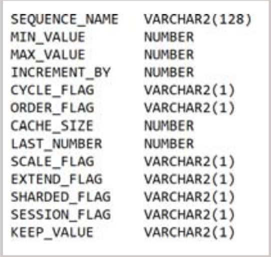


```
Sequence DEPT_DEPTID_SEQ altered.
```

Sequence Information

- The USER_SEQUENCES view describes all sequences that you own.

```
DESCRIBE user_sequences
```



```
SEQUENCE_NAME   VARCHAR2(128)
MIN_VALUE       NUMBER
MAX_VALUE       NUMBER
INCREMENT_BY    NUMBER
CYCLE_FLAG      VARCHAR2(1)
ORDER_FLAG      VARCHAR2(1)
CACHE_SIZE      NUMBER
LAST_NUMBER     NUMBER
SCALE_FLAG      VARCHAR2(1)
EXTEND_FLAG     VARCHAR2(1)
SHARDED_FLAG    VARCHAR2(1)
SESSION_FLAG    VARCHAR2(1)
KEEP_VALUE      VARCHAR2(1)
```

- Verify your sequence values in the USER_SEQUENCES data dictionary table.

```
SELECT  sequence_name, min_value, max_value,
increment_by, last_number
FROM user_sequences;
```

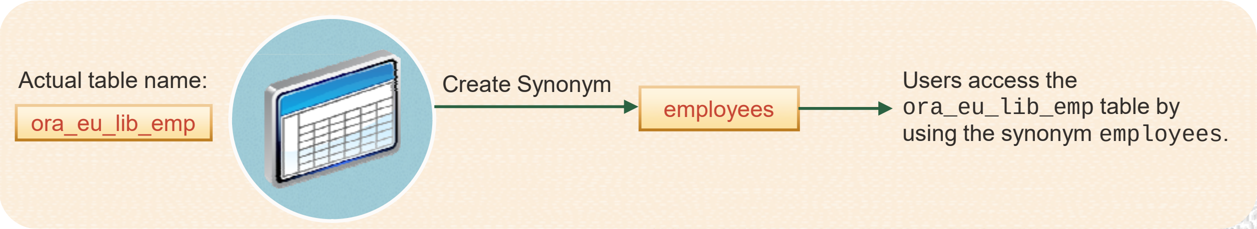
Lesson Agenda

- Overview of sequences
- Overview of synonyms
- Overview of indexes

Synonyms

A synonym:

- Is a database object
- Can be created to give an alternative name to a table or to another database object
- Requires no storage other than its definition in the data dictionary
- Is useful for hiding the identity and location of an underlying schema object



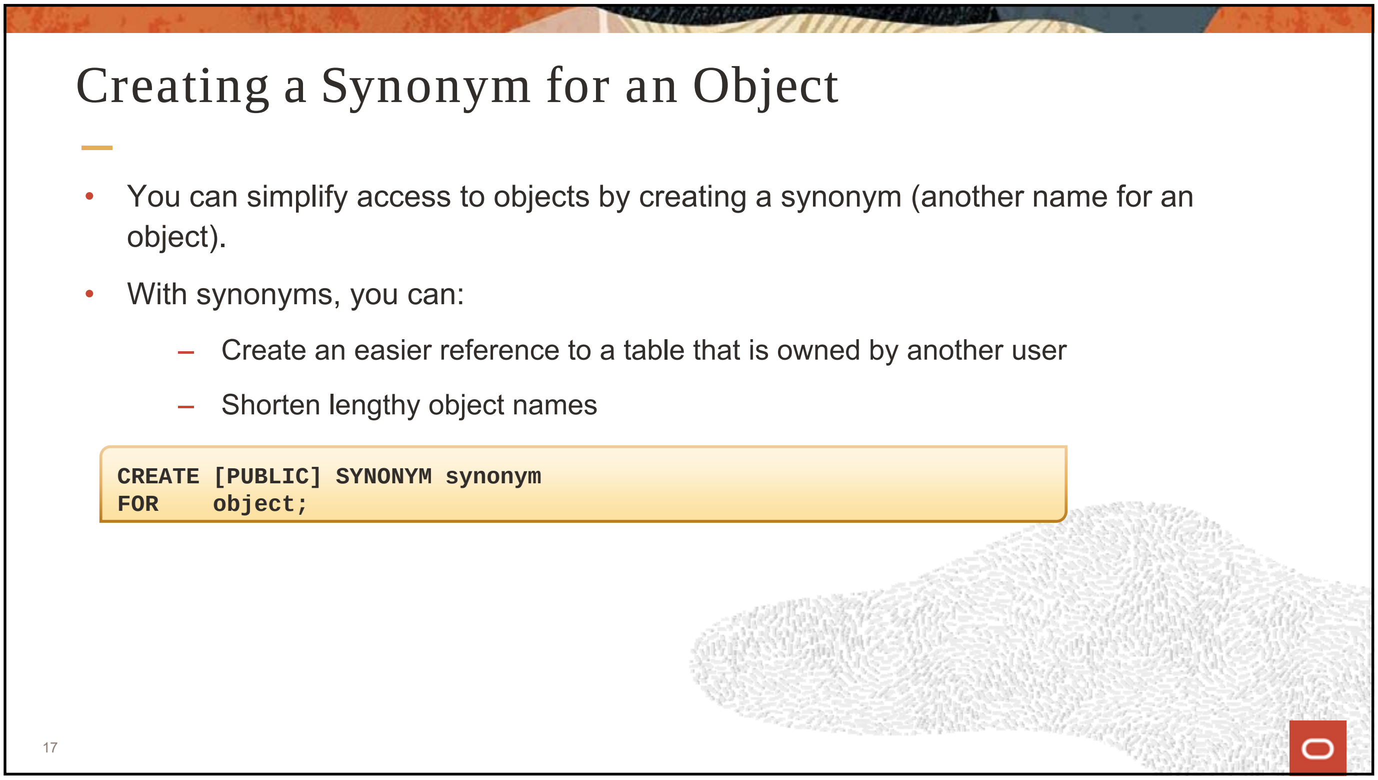
Actual table name:

ora_eu_lib_emp

Create Synonym

employees

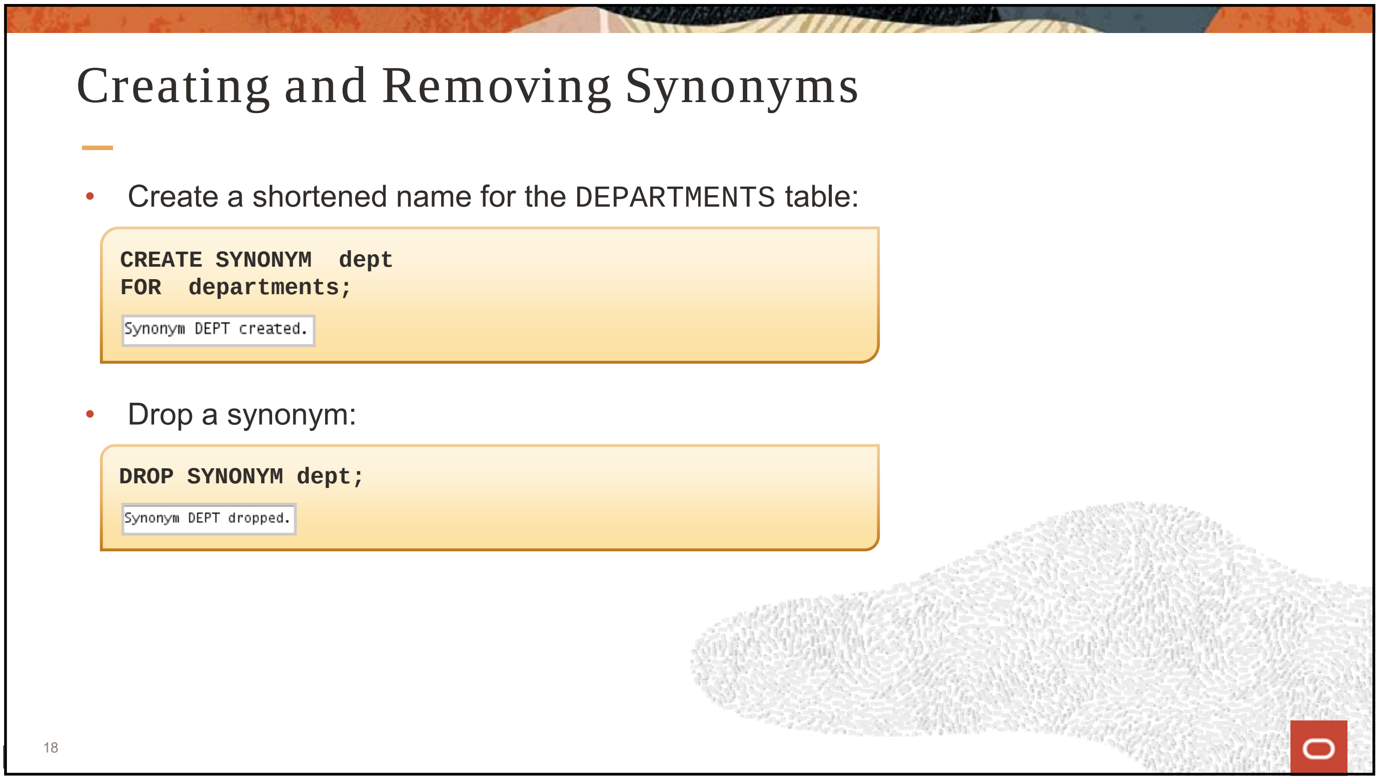
Users access the `ora_eu_lib_emp` table by using the synonym `employees`.



Creating a Synonym for an Object

- You can simplify access to objects by creating a synonym (another name for an object).
- With synonyms, you can:
  - Create an easier reference to a table that is owned by another user
  - Shorten lengthy object names

```
CREATE [PUBLIC] SYNONYM synonym
FOR    object;
```



Creating and Removing Synonyms

- Create a shortened name for the DEPARTMENTS table:

```
CREATE SYNONYM  dept
FOR  departments;
```

```
Synonym DEPT created.
```

- Drop a synonym:

```
DROP SYNONYM dept;
```

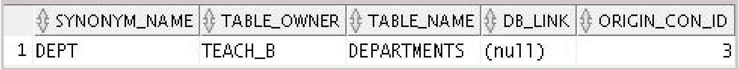
```
Synonym DEPT dropped.
```

Synonym Information

```
DESCRIBE user_synonyms
```

```
Name          Null     Type
------------- -------- -------------
SYNONYM_NAME  NOT NULL VARCHAR2(128)
TABLE_OWNER            VARCHAR2(128)
TABLE_NAME    NOT NULL VARCHAR2(128)
DB_LINK                VARCHAR2(128)
ORIGIN_CON_ID          NUMBER
```

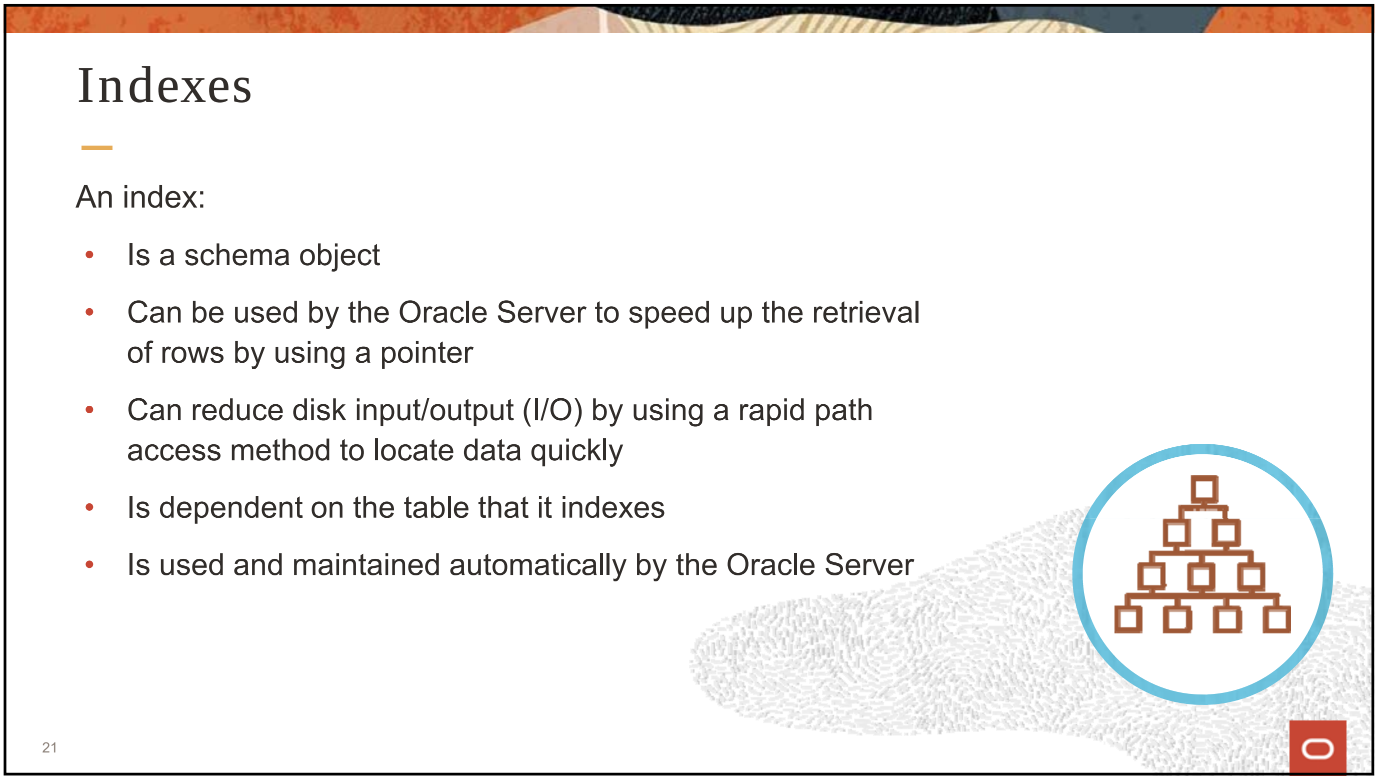
```
SELECT *
FROM   user_synonyms;
```



	SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK	ORIGIN_CON_ID
1	DEPT	TEACH_B	DEPARTMENTS	(null)	3

Lesson Agenda

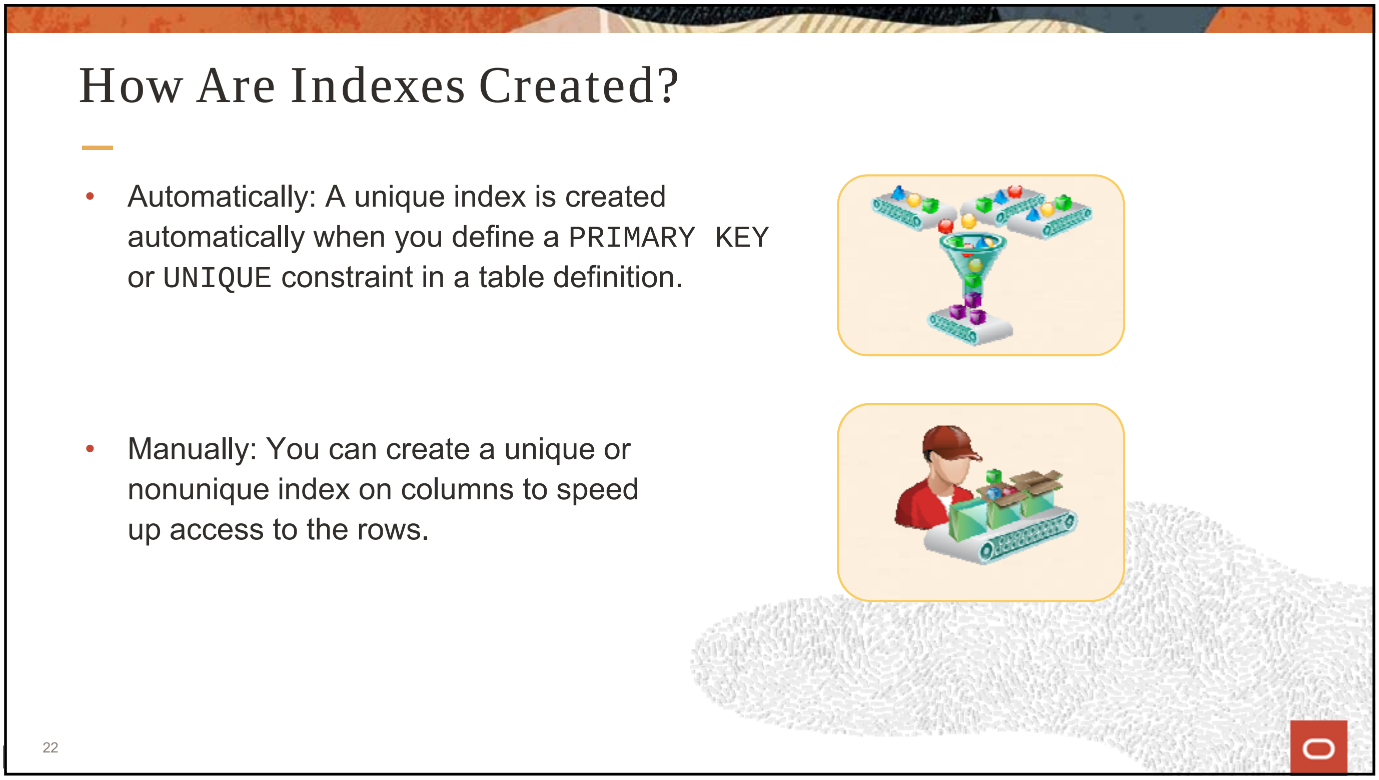
- Overview of sequences
- Overview of synonyms
- Overview of indexes



Indexes

An index:

- Is a schema object
- Can be used by the Oracle Server to speed up the retrieval of rows by using a pointer
- Can reduce disk input/output (I/O) by using a rapid path access method to locate data quickly
- Is dependent on the table that it indexes
- Is used and maintained automatically by the Oracle Server



How Are Indexes Created?

- Automatically: A unique index is created automatically when you define a `PRIMARY KEY` or `UNIQUE` constraint in a table definition.
- Manually: You can create a unique or nonunique index on columns to speed up access to the rows.

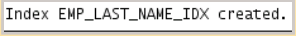
Creating an Index

- Create an index on one or more columns:

```
CREATE [UNIQUE]INDEX index
ON table (column[, column]...);
```

- Improve the speed of query access to the LAST_NAME column in the EMPLOYEES table:

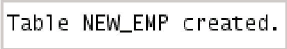
```
CREATE INDEX   emp_last_name_idx
ON employees(last_name);
```



```
Index EMP_LAST_NAME_IDX created.
```

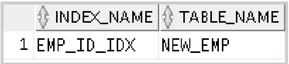
CREATE INDEX with the CREATE TABLE Statement

```
CREATE TABLE NEW_EMP
(employee_id NUMBER(6)
             PRIMARY KEY USING INDEX
            (CREATE INDEX emp_id_idx ON
            NEW_EMP(employee_id)),
first_name  VARCHAR2(20),
last_name   VARCHAR2(25));
```



```
Table NEW_EMP created.
```

```
SELECT INDEX_NAME, TABLE_NAME
FROM   USER_INDEXES
WHERE  TABLE_NAME = 'NEW_EMP';
```

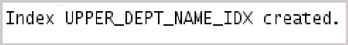


	INDEX_NAME	TABLE_NAME
1	EMP_ID_IDX	NEW_EMP

Function-Based Indexes

- A function-based index is based on expressions.
- The index expression is built from table columns, constants, SQL functions, and user-defined functions.

```
CREATE INDEX upper_last_name_idx
ON new_emp(UPPER(last_name));
```



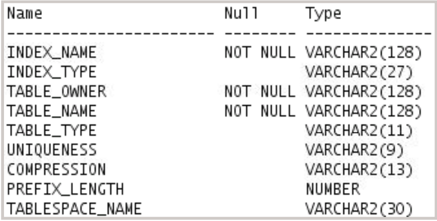
```
Index UPPER_DEPT_NAME_IDX created.
```

```
SELECT *
FROM   new_emp
WHERE  UPPER(last_name) = 'KING';
```

Index Information

- USER_INDEXES provides information about your indexes.
- USER_IND_COLUMNS describes columns of indexes owned by you and columns of indexes on your tables.

```
DESCRIBE user_indexes
```



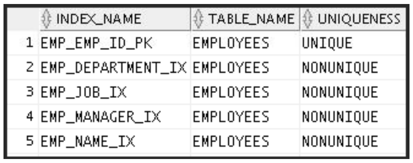
```
Name                    Null     Type
----------------------- -------- --------------
INDEX_NAME              NOT NULL VARCHAR2(128)
INDEX_TYPE                       VARCHAR2(27)
TABLE_OWNER             NOT NULL VARCHAR2(128)
TABLE_NAME              NOT NULL VARCHAR2(128)
TABLE_TYPE                       VARCHAR2(11)
UNIQUENESS                       VARCHAR2(9)
COMPRESSION                      VARCHAR2(13)
PREFIX_LENGTH                    NUMBER
TABLESPACE_NAME                  VARCHAR2(30)
```

...

USER_INDEXES: Examples

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

1

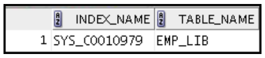


	INDEX_NAME	TABLE_NAME	UNIQUENESS
1	EMP_EMP_ID_PK	EMPLOYEES	UNIQUE
2	EMP_DEPARTMENT_IX	EMPLOYEES	NONUNIQUE
3	EMP_JOB_IX	EMPLOYEES	NONUNIQUE
4	EMP_MANAGER_IX	EMPLOYEES	NONUNIQUE
5	EMP_NAME_IX	EMPLOYEES	NONUNIQUE

...

```
SELECT index_name, table_name
FROM   user_indexes
WHERE  table_name = 'EMP_LIB';
```

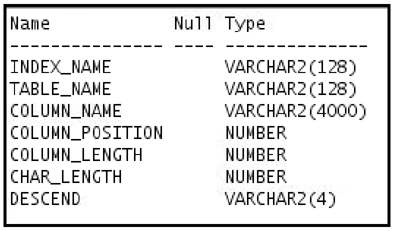
2



	INDEX_NAME	TABLE_NAME
1	SYS_C0010979	EMP_LIB

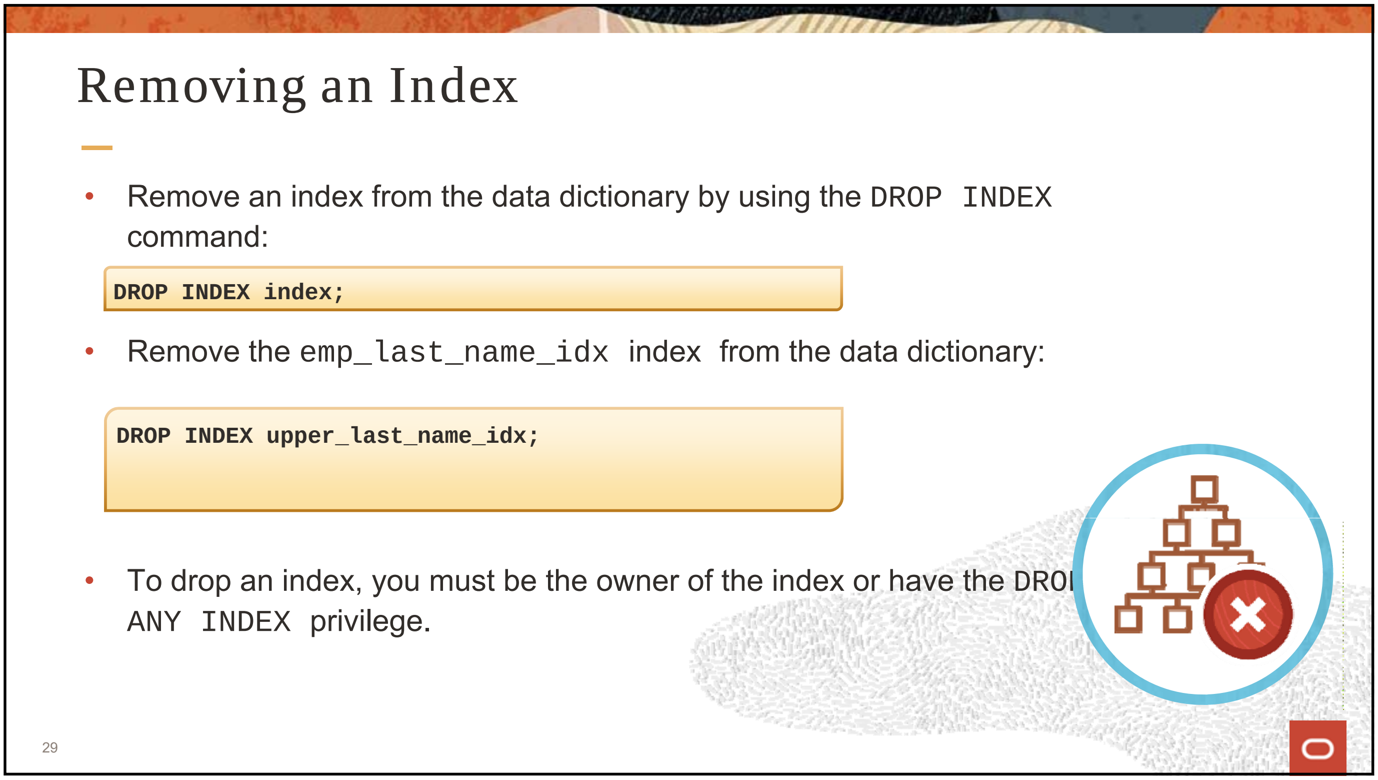
Querying USER_IND_COLUMNS

```
DESCRIBE user_ind_columns
```



```
Name            Null Type
--------------- ---- --------------
INDEX_NAME           VARCHAR2(128)
TABLE_NAME           VARCHAR2(128)
COLUMN_NAME          VARCHAR2(4000)
COLUMN_POSITION      NUMBER
COLUMN_LENGTH        NUMBER
CHAR_LENGTH          NUMBER
DESCEND              VARCHAR2(4)
```

```
SELECT index_name, column_name, table_name
FROM   user_ind_columns
WHERE  index_name = 'EMP_JOB_IX';
```



Removing an Index

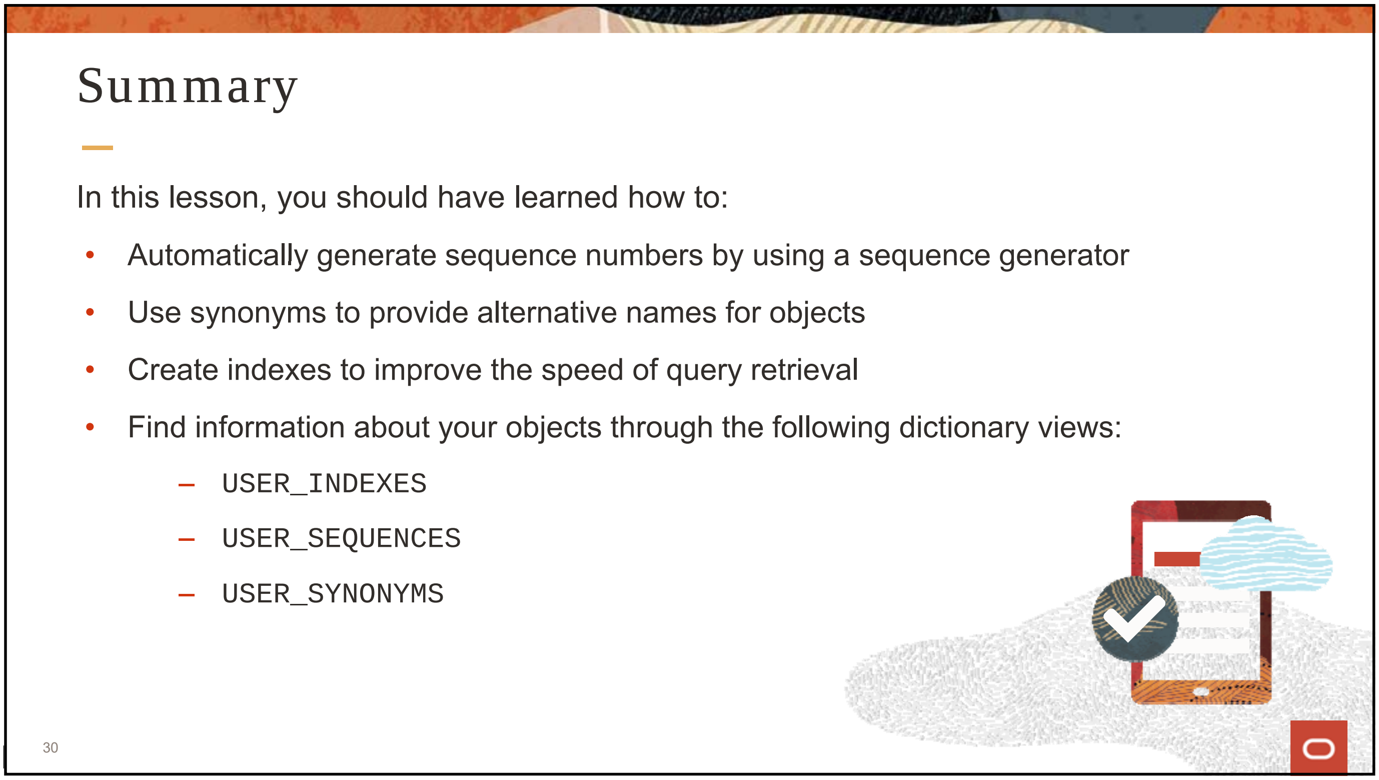
- Remove an index from the data dictionary by using the `DROP INDEX` command:

```
DROP INDEX index;
```

- Remove the `emp_last_name_idx` index from the data dictionary:

```
DROP INDEX upper_last_name_idx;
```

- To drop an index, you must be the owner of the index or have the `DROP ANY INDEX` privilege.



Summary

In this lesson, you should have learned how to:

- Automatically generate sequence numbers by using a sequence generator
- Use synonyms to provide alternative names for objects
- Create indexes to improve the speed of query retrieval
- Find information about your objects through the following dictionary views:
  - `USER_INDEXES`
  - `USER_SEQUENCES`
  - `USER_SYNONYMS`