U2 SQL

Friday, October 29, 2021 14:23

1. What are the DDL statements? Explain with their syntax.

a. Statement used to define a database structure or schema

DDL Statements	Description	Syntax	eg
CREATE	to create a new database object	CREATE TABLE table_name (attribute_name data_type [(size)] [constraint],….)	CREATE TABLE PERSON (PID NUMBER PRIMARY KEY, PNAME VARCHAR2(50), PADDRESS VARCHAR2(50), AadharID NUMBER(12), MOBILENO NUMBER(10))

ALTER	• to change an aspect of the structure of an existing database object
	• to add, delete, modify the attributes of the relations (tables) in the database.
		// 1. add a column to the existing table
		ALTER TABLE tableName
		//2. drop a column from the existing table
		ALTER TABLE tableName
		//3. rename a column in the existing table
		ALTER TABLE tableName
		//4. modify the datatype of an already existing column in the table
		ALTER TABLE table_name
			// 1. add a column to the existing table
			ADD columnName columnDefinition;
			//2. drop a column from the existing table
			DROP COLUMN columnName;
			//3. rename a column in the existing table
			RENAME COLUMN olderName TO newName;
			//4. modify the datatype of an already existing column in the table
			ALTER COLUMN column_name column_type;

DROP	to drop (remove) a	DROP <OBJECT>	DROP TABLE STUDENTS

	database object	<object name>	
TRUNCATE	remove all records from a table, including all spaces allocated for the records are removed	TRUNCATE TABLE <table name>	TRUNCATE TABLE STUDENTS
RENAME	rename an database object	RENAME <old object name> TO <new object name>	

2. What are the DML statements? Explain with their syntax.

a. DML (Data Manipulation Language) statements are the element in the SQL language that is used for data retrieval and manipulation.

b.

DML Statements	Description	Syntax	eg
INSERT	• to add rows to a table • Insert data into a relation. • insert statement is a request to insert one tuple	insert into t values (val1, val2, ..., valn); OR insert into t(A1, A2,..., An) values (val1, val2, ..., valn);	INSERT INTO departments VALUES (280, 'Recreation', 121, 1700);
UPDATE	• to change column values of existing rows • for modifying existing tuples in a table	UPDATE table SET attr1 =val1, attr2=val2, ... [WHERE condition];	UPDATE employees SET salary = salary + 1000.0;
DELETE	• to remove rows from a table • to delete a one or more records from a table	DELETE FROM table [WHERE conditions];	DELETE FROM suppliers WHERE supplier_name = 'IBM';

3. What are the different data types used in Oracle SQL?

a. d
b. D

	Data Type	Description	
	char(n)	A fixed-length character string with user-specified length n.	default is 1 byte
	varchar(n)	A variable-length character string with user-specified maximum length n	
	Varchar2(n)		

	int/ integer:	An integer.	
	number(p, d):	A fixed-point number with user-specified precision. The number consists of p digits (plus a sign), and d of the p digits are to the right of the decimal point.	Stores fixed and floating point numbers up to 38 digits of precision
	float(n):	A floating-point number.	
	real, double precision:	Floating-point and double-precision floating-point numbers with machine-dependent precision.	
	date:	A calendar date containing a (four-digit) year, month, and day of the month. Format: YYYY-MM-DD	
	time:	The time of day, in hours, minutes, and seconds. Format: hh:mm:ss	
	timestamp:	A combination of date and time. Format: YYYY-MM-DD hh:mm:ss	

c. d

4. Explain different types of constraints on attributes in relational database management system.

a. D

b. D

	SQL Constraints	Description
	NOT NULL	Ensures that a column cannot have a NULL value
	UNIQUE	Ensures that all values in a column are different
	PRIMARY KEY	A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
	FOREIGN KEY	Uniquely identifies a row/record in another table
	CHECK	Ensures that all values in a column satisfies a specific condition
	DEFAULT	Sets a default value for a column when no value is specified
	INDEX	Used to create and retrieve data from the database very quickly

c. Not Null Constraint

   i. If a column in a table is specified as Not Null,

   ii. then it’s not possible to insert a null in such column.

   iii. It can be implemented with create and alter commands.

   iv. When we implement the Not Null constraint with alter command there should not be any null values in the existing table.

d. Unique Constraint

   i. The unique constraint doesn’t allow duplicate values in a column.

   ii. If unique constraint encompasses two or more columns, no two equal combinations are allowed

e. DEFAULT Constraint

   i. The DEFAULT constraint is used to insert a default value into a column.

   ii. The default value will be added to all new records, if no other value is specified.

f. Primary Key Constraints

i. The PRIMARY KEY constraint uniquely identifies each record in a table.
ii. Primary keys must contain UNIQUE values, and cannot contain NULL values
iii. A table can have only ONE primary key; and in the table, this primary key can consist of single or multiple columns (fields).

g. FOREIGN Key Constraints

i. A FOREIGN KEY is a key used to link two tables together.
ii. A FOREIGN KEY is a field (or collection of fields) in one table that refers to the PRIMARY KEY in another table.
iii. The table containing the foreign key is called the child table, and the table containing the candidate key is called the referenced or parent table.

h. CHECK Constraint

i. CHECK constraint is used to limit the value range that can be placed in a column.
ii. If you define a CHECK constraint on a single column it allows only certain values for this column.
iii. If you define a CHECK constraint on a table it can limit the values in certain columns based on values in other columns in the row.

5. Demonstrate the use of alter command with examples.

a. The SQL ALTER TABLE command is used to add, delete or modify columns in an existing table.

b. to add and drop various constraints on an existing table.

c. Syntax:

d. The basic syntax of ALTER TABLE to add a new column in an existing table is as follows:

```
ALTER TABLE table_name ADD column_name datatype;
```

e. The basic syntax of ALTER TABLE to DROP COLUMN in an existing table is as follows:

```
ALTER TABLE table_name DROP COLUMN column_name;
```

f. The basic syntax of ALTER TABLE to change the DATA TYPE of a column in a table is as follows:

```
ALTER TABLE table_name MODIFY column_name datatype;
```

g.

Syntax:	ALTER TABLE <table name> ADD CONSTRAINT <constraint name> <constraint type> <column name>; ALTER TABLE <table name> ADD CONSTRAINT <constraint name> FOREIGN KEY (<foreign key column>) REFERENCES <parent table name> (<primary key column>);

Example	ALTER TABLE Employee ADD CONSTRAINT c_name NOT NULL Ename;
	ALTER TABLE Employee ADD CONSTRAINT Ref_dept FOREIGN KEY(deptno) REFERENCES Department(deptno);

h.

6. Write SQL Statements to perform following tasks

a. To add new column in existing table.

i. ALTER TABLE command can be used to add, delete or modify columns in an existing table

ii. ALTER TABLE - ADD Column
To add a column in a table, use the following syntax:

```
ALTER TABLE table_name
ADD column_name datatype;
```

b. To remove column from existing table.

i. ALTER TABLE - DROP COLUMN

To delete a column in a table, use the following syntax

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

c. To change data type of column in existing table.

To change the data type of a column in a table, use the following syntax:

SQL Server / MS Access:

```
ALTER TABLE table_name
ALTER COLUMN column_name datatype;
```

My SQL / Oracle (prior version 10G):

```
ALTER TABLE table_name
MODIFY COLUMN column_name datatype;
```

Oracle 10G and later:

```
ALTER TABLE table_name
MODIFY column_name datatype;
```

d. To add NOT NULL constraint on particular attribute.

The following is the syntax to add a constraint to an existing column.

```
ALTER table yourTableName  modify column_name data type
constraint;
```

To add the NOT NULL constraint

```
ALTER table AddNotNUlldemo modify name varchar(100) not null;
```

e. To add check constraint on particular attribute.

f. To add primary key to table.

```
ALTER TABLE table_name
  ADD CONSTRAINT constraint_name
    PRIMARY KEY (column1, column2, ... column_n);
```

g. To add foreign key to table.

7. Give three variations of insert command with example.

8. What are the different types of SQL joins?

a. Cartesian Product
b. Inner Join
c. Left outer join
d. Right outer join
e. Full outer join
f. Natural join

9. Explain different variations of Outer Join – full, right, left with example

10. Explain inner join with example

a. Retrieve the records having same value for similar attributes in two tables

b. Use a comparison operator to match rows from two tables

c. Retrieve the data based on the values in common columns from each table.

d. When to use inner join

i. Use an inner join when you want to match values from both tables.

e. Why to use Inner Joins:

i. Use inner joins to obtain information from two separate tables and combine that information in one result set.

ii. Eg

1) SELECT FROM INNER JOIN ON table1.attribute=table2.attribute

2) SELECT * FROM PERSON INNER JOIN STUDENT ON PERSON.PID=STUDENT.PID
SELECT <list_of_attributes> FROM
<Table1> INNER JOIN <Table2>
USING (<attribute>)

3) SELECT * FROM PERSON INNER JOIN
STUDENT USING (PID)

4) The columns listed in the USING clause must be present in both of the two tables being joined.

5) The USING clause will be transformed to an ON clause that checks for equality between the named columns in the two tables.

11. Explain Cartesian product/ Cross Join with example

a. When we join every row of a table to every row of another table we get Cartesian Product

b. Returns All rows from first table, Each row from the first table is combined with all rows from the second table

c. Cross join – every row of one table is matched with every row of another table.

d. Cartesian join and cross join are same

e. If T1 and T2 are two sets then

f. Number of records in T1 CROSS JOIN T2 are

g. (No. of records in T1) X (No. of records in T2)

h. It does not check for common attribute values.

i. T1 X T2

j. Select * from Table1,Table2;

k. SELECT * FROM EMP CROSS JOIN DEPT;

l. SELECT * FROM EMP, DEPT;

m. In first statement it is specified explicitly

n. Second one is implicit

o. E.g.

i. SELECT <list_of_attributes> FROM <Table1>, <Table2>…

▪ SELECT * FROM PERSON, STUDENT

ii. SELECT <list_of_attributes> FROM <Table1> CROSS JOIN <Table2>

▪ SELECT * FROM PERSON CROSS JOIN STUDENT

p.

q.

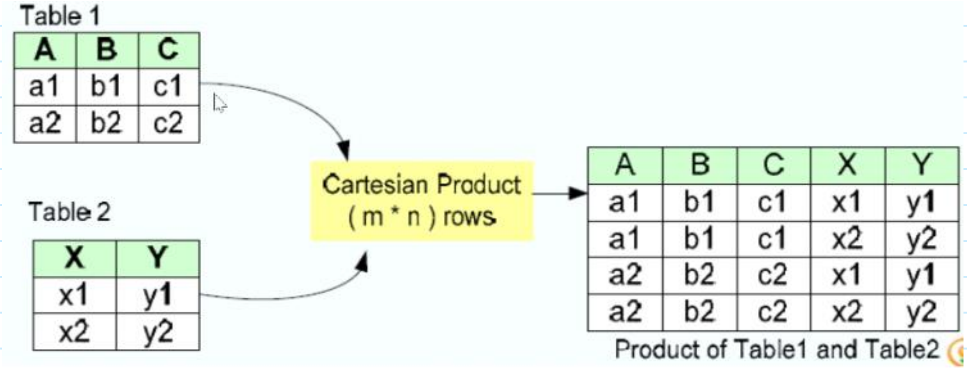

Table 1

A	B	C
a1	b1	c1
a2	b2	c2

Table 2

X	Y
x1	y1
x2	y2

A	B	C	X	Y
a1	b1	c1	x1	y1
a1	b1	c1	x2	y2
a2	b2	c2	x1	y1
a2	b2	c2	x2	y2

Product of Table1 and Table2

r. g

12. Explain natural join with example

a. Similar to Inner Join

b. Retrieve the records having same value for similar attributes in two tables.

c. takes condition implicitly unlike the inner join which takes conditions explicitly

d. Column names in both tables must be same for natural join

e. Syntax

```
SELECT FROM NATURAL JOIN
```

f. Eg

```
SELECT * FROM FACULTY NATURAL JOIN STUDENT
```

13. Explain different aggregate functions with example

- Aggregate functions are functions that take a collection (a set or multiset) of values as input and return a single value
- Aggregate functions are used in place of column names in the SELECT statement

i. MIN

1) Returns the smallest value that occurs in the specified column.
2) Column need not be numeric type.
3) MIN ignores any null values.
4) `SELECT MIN( column name/ expression) FROM Table_name;`
5) Selecting Minimum account balance;

```
SELECT Min(balance) as MinimumBalance FROM account;
```

ii. MAX

1) Returns the largest value that occurs in the specified column.
2) Column need not be numeric type.
3) MAX ignores any null values.
4) `SELECT MAX( column name/ expression) FROM Table_name;`
5) Selecting Maximum loan amount

```
SELECT MAX(amount) as MaximumAmount FROM loan;
```

iii. AVERAGE

1) Returns the average of all the values in the specified column.
2) SQL AVG() ignores Null Values.
3) Column must be numeric data type

4)

Syntax	E.G.
`AVG ( [ DISTINCT ] column-name/ expression )`	List the average account balance of customers `SELECT AVG(balance) as "Average Bal" FROM account;`

iv. SUM

1) returns the sum of numeric column.
2) SQL SUM() ignores Null Values
3)

Syntax	E.G.
`SUM ( [ DISTINCT ] column-name / expression )`	Find the sum of loan amount of bank `SELECT SUM(amount) as "Totalamount" FROM loan;`

v. COUNT

1) returns the number of tuples returned by the query as a number.
2)

Syntax	E.G.

`COUNT( [DISTINCT] column-name/ expression )`	List the total number of customers
	`SELECT COUNT(*) FROM customer;`
	
	List the total number of account holder at SBI Branch
	`SELECT COUNT(*) FROM account WHERE branch-name=‘SBI’;`
	
	List the total number of unique customer city
	`SELECT COUNT(DISTINCT customer-city) FROM customer;`

3) Count(*) = No of rows
4) Count(ColumnName) = No. of rows that do not have NULL Value

14. Explain the SQL Statements – like, in, between

15. Explain group by- having clause with example

16. Explain different set operations with example– union all, union, minus, intersect.

a. Oracle SQL supports following four set operations:

b. union all

i. Combines the results of two SELECT statements into one result set and it retain all duplicates

c. union

i. Combines the results of two SELECT statements into one result set, and then eliminates any duplicate rows from that result set.
ii. Union is like an “OR” operation

d. minus

i. Takes the result set of one SELECT statement, and removes those rows that are also returned by a second SELECT statement.
ii. It retrieves rows which are present in 1 but not in 2

e. Intersect

i. Returns only those rows that are returned by each of two SELECT statements.
ii. It retrieves those tuples which are present in both relation
iii. An intersection is an AND operation