1. **What is Database?**

A database is a collection of information that is organized so that it can easily be accessed, managed, and updated. In one view, databases can be classified according to types of content: bibliographic, full-text, numeric, and images.

1. **What is a Table?**

In relational databases and flat file databases, a table is a set of data elements (values) using a model of vertical columns (identifiable by name) and horizontal rows, the cell being the unit where a row and column intersect. A table has a specified number of columns, but can have any number of rows.

1. **What is a column?**

In the context of a relational database, a column is a set of data values of a particular simple type, one for each row of the table. The columns provide the structure according to which the rows are composed.

1. **What is a row?**

A row—also called a record or tuple—represents a single, implicitly structured data item in a table. In simple terms, a database table can be thought of as consisting of rows and columns or fields.

1. **Example for Inner join?**

**CUSTOMER TABLE:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **NAME** | **AGE** | **ADDRESS** | **SALARY** |
| 1 | Ramesh | 32 | Ahmedabad | 2000 |
| 2 | Harish | 25 | Hyderabad | 1500 |
| 3 | Krishna | 23 | Delhi | 2000 |
| 4 | Phani | 25 | Guntur | 6500 |
| 5 | Mohan | 27 | Vizag | 8500 |
| 6 | Subha | 22 | Orissa | 4500 |
| 7 | Divya | 24 | Mumbai | 10000 |

**ORDER TABLE:**

|  |  |  |  |
| --- | --- | --- | --- |
| **OID** | **DATE** | **CUSTOMER\_ID** | **AMOUNT** |
| 102 | 2009-10-08 | 3 | 3000 |
| 100 | 2009-10-08 | 3 | 1500 |
| 101 | 2009-11-20 | 2 | 1560 |
| 103 | 2008-05-20 | 4 | 2060 |

**JOIN TWO TABLES USING INNER JOIN:**

SQL> SELECT ID, NAME, AMOUNT, DATE

FROM CUSTOMERS

INNER JOIN ORDERS

ON CUSTOMERS.ID = ORDERS.CUSTOMER\_ID;

**RESULT:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **AMOUNT** | **DATE** |
| 3 | Krishna | 3000 | 2009-10-08 |
| 3 | Krishna | 1500 | 2009-10-08 |
| 2 | Harish | 1560 | 2009-11-20 |
| 4 | Phani | 2060 | 2008-05-20 |

1. **Example for Left Join?**

**Join two tables using LEFT JOIN as follows:**

SQL> SELECT ID, NAME, AMOUNT, DATE

FROM CUSTOMERS

LEFT JOIN ORDERS

ON CUSTOMERS.ID = ORDERS.CUSTOMER\_ID;

**Result:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **AMOUNT** | **DATE** |
| 1 | Ramesh | NULL | NULL |
| 2 | Harish | 1560 | 2009-11-20 |
| 3 | Krishna | 3000 | 2009-10-08 |
| 3 | Krishna | 1500 | 2009-10-08 |
| 4 | Phani | 2060 | 2008-05-20 |
| 5 | Mohan | NULL | NULL |
| 6 | Subha | NULL | NULL |
| 7 | Divya | NULL | NULL |

1. **Example for Right Outer join?**

**Syntax:**

SQL> SELECT ID, NAME, AMOUNT, DATE

FROM CUSTOMERS

RIGHT JOIN ORDERS

ON CUSTOMERS.ID = ORDERS.CUSTOMER\_ID;

**Result:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **AMOIUNT** | **DATE** |
| 3 | Krishna | 3000 | 2009-10-08 |
| 3 | Krishna | 1500 | 2009-10-08 |
| 2 | Harish | 1560 | 2009-11-20 |
| 4 | Phani | 2060 | 2008-05-20 |

1. **Example for Max, sun, Avg?**

**Max Syntax:**

SELECT MAX(column-name)

FROM table-name

**Example:**

SELECT MAX(TotalAmount)

FROM [Order]

WHERE YEAR(OrderDate) = 2014

----------------------------

**Sum Syntax:**

SELECT SUM(column-name)

FROM table-name

**Example:**

SELECT AVG(column-name)

FROM table-name

------------------------------

**Avg Syntax:**

SELECT AVG(column-name)

FROM table-name

**Example:**

SELECT AVG(TotalAmount)

FROM [Order]

1. **Example for GROUP BY?**

**SQL Syntax:**

SELECT column\_name, aggregate\_function (column\_name)  
FROM table\_name  
WHERE column\_name operator value  
GROUP BY column\_name;

**Example:**

SELECT Shippers.ShipperName,COUNT(Orders.OrderID) AS NumberOfOrders FROM Orders

LEFT JOIN Shippers

ON Orders.ShipperID=Shippers.ShipperID

GROUP BY ShipperName;

1. **Example for Having?**

**Syntax:**

SELECT

FROM

WHERE

GROUP BY

HAVING

ORDER BY

**Example:**

Consider the Customer table:

SQL > SELECT ID, NAME, AGE, ADDRESS, SALARY

FROM CUSTOMERS

GROUP BY age

HAVING COUNT(age) >= 2;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **NAME** | **AGE** | **ADDRESS** | **SALARY** |
| 2 | Harish | 25 | Hyderabad | 1500 |

1. **Example for Where Condition?**

**Syntax:**

SELECT *column\_name*,*column\_name*  
FROM *table\_name*  
WHERE *column\_name operator value*;

**Example:**

SELECT \* FROM Customers  
WHERE Country='Mexico';

1. **Example for Primary Key?**

**Syntax:**

CREATE TABLE CUSTOMERS(

ID INT NOT NULL,

NAME VARCHAR (20) NOT NULL,

AGE INT NOT NULL,

ADDRESS CHAR (25) ,

SALARY DECIMAL (18, 2),

PRIMARY KEY (ID)

**PRIMARY KEY constraint on multiple columns, use this syntax:**

CREATE TABLE CUSTOMERS(

ID INT NOT NULL,

NAME VARCHAR (20) NOT NULL,

AGE INT NOT NULL,

ADDRESS CHAR (25) ,

SALARY DECIMAL (18, 2),

PRIMARY KEY (ID, NAME)

);

1. **Example for Foreign Key?**

**CUSTOMERS Table:**

CREATE TABLE CUSTOMERS(

ID INT NOT NULL,

NAME VARCHAR (20) NOT NULL,

AGE INT NOT NULL,

ADDRESS CHAR (25) ,

SALARY DECIMAL (18, 2),

PRIMARY KEY (ID)

);

**ORDERS Table:**

CREATE TABLE ORDERS (

ID INT NOT NULL,

DATE DATETIME,

CUSTOMER\_ID INT references CUSTOMERS(ID),

AMOUNT double,

PRIMARY KEY (ID)

);

1. **Find second highest salary from row table?**

SELECT MAX(Salary) FROM Employee

WHERE Salary NOT IN (SELECT MAX(Salary) FROM Employee )