DECODE and CASE statements in Oracle both provide a conditional construct, of this form:

if A = n1 then A1  
else if A = n2 then A2  
else X

Databases before Oracle 8.1.6 had only the DECODE function. CASE was introduced in Oracle 8.1.6 as a standard, more meaningful and more powerful function.

Everything DECODE can do, CASE can. There is a lot else CASE can do though, which DECODE cannot. We’ll go through detailed examples in this article.

SQL> select ename

2 , decode (deptno, 10, 'Accounting',

3 20, 'Research',

4 30, 'Sales',

5 'Unknown') as department

6 fromemp

7 whererownum< 4;

ENAME DEPARTMENT

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

SMITH Research

ALLEN Sales

WARD Sales

SQL> select ename

2 , case deptno

3 when 10 then 'Accounting'

4 when 20 then 'Research'

5 when 30 then 'Sales'

6 else 'Unknown'

7 end as department

8 fromemp

9 whererownum< 4;

ENAME DEPARTMENT

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

SMITH Research

ALLEN Sales

WARD Sales

Complicated logical comparisons in DECODE, even if technically achievable, are a recipe for messy, bug-prone code. When the same can be done more cleanly with CASE, go for CASE.

**1. CASE can work with logical operators other than ‘=’**

DECODE performs an equality check only. CASE is capable of other logical comparisons such as <> etc. It takes some complex coding – forcing ranges of data into discrete form – to achieve the same effect with DECODE.

An example of putting employees in grade brackets based on their salaries. This can be done elegantly with CASE.

SQL> select ename

2 , case

3 when sal< 1000

4 then 'Grade I'

5 when (sal>=1000 and sal< 2000)

6 then 'Grade II'

7 when (sal>= 2000 and sal< 3000)

8 then 'Grade III'

9 else 'Grade IV'

10 end sal\_grade

11 fromemp

12 whererownum< 4;

ENAME SAL\_GRADE

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

SMITH Grade I

ALLEN Grade II

WARD Grade II

**2. CASE can work with predicates and searchable subqueries**

DECODE works with expressions that are scalar values only. CASE can work with predicates and subqueries in searchable form.

An example of categorizing employees based on reporting relationship, showing these two uses of CASE.

SQL> select e.ename,

2 case

3 -- predicate with "in"

4 -- set the category based on ename list

5 when e.ename in ('KING','SMITH','WARD')

6 then 'Top Bosses'

7 -- searchable subquery

8 -- identify if this emp has a reportee

9 when exists (select 1 from emp emp1

10 where emp1.mgr = e.empno)

11 then 'Managers'

12 else

13 'General Employees'

14 end emp\_category

15 fromemp e

16 whererownum< 5;

ENAME EMP\_CATEGORY

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

SMITH Top Bosses

ALLEN General Employees

WARD Top Bosses

JONES Managers

**3. CASE can work as a PL/SQL construct**

DECODE can work as a function inside SQL only. CASE can be an efficient substitute for IF-THEN-ELSE in PL/SQL.

SQL> declare

2 grade char(1);

3 begin

4 grade := 'b';

5 case grade

6 when 'a' then dbms\_output.put\_line('excellent');

7 when 'b' then dbms\_output.put\_line('very good');

8 when 'c' then dbms\_output.put\_line('good');

9 when 'd' then dbms\_output.put\_line('fair');

10 when 'f' then dbms\_output.put\_line('poor');

11 else dbms\_output.put\_line('no such grade');

12 end case;

13 end;

14 /

PL/SQL procedure successfully completed.

CASE can even work as a parameter to a procedure call, while DECODE cannot.

SQL>var a varchar2(5);

SQL>exec :a := 'THREE';

PL/SQL procedure successfully completed.

SQL>

SQL> create or replace procedure proc\_test (i number)

2 as

3 begin

4 dbms\_output.put\_line('output = '||i);

5 end;

6 /

Procedure created.

SQL> exec proc\_test(decode(:a,'THREE',3,0));

BEGIN proc\_test(decode(:a,'THREE',3,0)); END;

\*

ERROR at line 1:

ORA-06550: line 1, column 17:

PLS-00204: function or pseudo-column 'DECODE' may be used inside a SQL

statement only

ORA-06550: line 1, column 7:

PL/SQL: Statement ignored

SQL> exec proc\_test(case :a when 'THREE' then 3 else 0 end);

output = 3

PL/SQL procedure successfully completed.

**4. Careful! CASE handles NULL differently**

Check out the different results with DECODE vs NULL.

SQL> select decode(null

2 , null, 'NULL'

3 , 'NOT NULL'

4 ) null\_test

5 from dual;

NULL

----

NULL

SQL> select case null

2 when null

3 then 'NULL'

4 else 'NOT NULL'

5 end null\_test

6 from dual;

NULL\_TES

--------

NOT NULL

The “[searched CASE](http://www.oratable.com/simple-case-searched-case/)” works as does DECODE.

SQL> select case

2 when null is null

3 then 'NULL'

4 else 'NOT NULL'

5 end null\_test

6\* from dual

SQL> /

NULL\_TES

--------

NULL

**5. CASE expects datatype consistency, DECODE does not**

Compare the two examples below- DECODE gives you a result, CASE gives a datatype mismatch error.

SQL> select decode(2,1,1,

2 '2','2',

3 '3') t

4 from dual;

T

----------

2

SQL> select case 2 when 1 then '1'

2 when '2' then '2'

3 else '3'

4 end

5 from dual;

when '2' then '2'

\*

ERROR at line 2:

ORA-00932: inconsistent datatypes: expected NUMBER got CHAR

**6. CASE is ANSI SQL-compliant**

CASE complies with ANSI SQL. DECODE is proprietary to Oracle.

**7. The difference in readability**

In very simple situations, DECODE is shorter and easier to understand than CASE.

SQL> -- An example where DECODE and CASE

SQL> -- can work equally well, and

SQL> -- DECODE is cleaner

**Cursors**

Oracle creates a memory area, known as context area, for processing an SQL statement, which contains all information needed for processing the statement, for example, number of rows processed, etc.

A cursor is a pointer to this context area. PL/SQL controls the context area through a cursor. A cursor holds the rows (one or more) returned by a SQL statement. The set of rows the cursor holds is referred to as the **active set**.

You can name a cursor so that it could be referred to in a program to fetch and process the rows returned by the SQL statement, one at a time. There are two types of cursors:

* Implicit cursors
* Explicit cursors

## Implicit Cursors

Implicit cursors are automatically created by Oracle whenever an SQL statement is executed, when there is no explicit cursor for the statement. Programmers cannot control the implicit cursors and the information in it.

Whenever a DML statement (INSERT, UPDATE and DELETE) is issued, an implicit cursor is associated with this statement. For INSERT operations, the cursor holds the data that needs to be inserted. For UPDATE and DELETE operations, the cursor identifies the rows that would be affected.

In PL/SQL, you can refer to the most recent implicit cursor as the **SQL cursor**, which always has the attributes like %FOUND, %ISOPEN, %NOTFOUND, and %ROWCOUNT. The SQL cursor has additional attributes, %BULK\_ROWCOUNT and %BULK\_EXCEPTIONS, designed for use with the FORALL statement. The following table provides the description of the most used attributes:

%FOUND: Returns TRUE if an INSERT, UPDATE, or DELETE statement affected one or more rows or a SELECT INTO statement returned one or more rows. Otherwise, it returns FALSE.

%NOTFOUND: The logical opposite of %FOUND. It returns TRUE if an INSERT, UPDATE, or DELETE statement affected no rows, or a SELECT INTO statement returned no rows. Otherwise, it returns FALSE.

%ISOPEN: Always returns FALSE for implicit cursors, because Oracle closes the SQL cursor automatically after executing its associated SQL statement.

%ROWCOUNT: Returns the number of rows affected by an INSERT, UPDATE, or DELETE statement, or returned by a SELECT INTO statement.

Any SQL cursor attribute will be accessed as **sql%attribute\_name** as shown below in the example.

## Example:

We will be using the CUSTOMERS table we had created and used in the previous chapters.

Select\*from customers;

+--+-----+-----+--------+-------+

|ID| NAME| AGE | ADDRESS| SALARY|

+--+-----+-----+--------+-------+

|1|Ramesh|32|Ahmedabad|2000.00|

|2|Khilan|25|Delhi|1500.00|

|3|kaushik|23|Kota|2000.00|

|4|Chaitali|25|Mumbai|6500.00|

|5|Hardik|27|Bhopal|8500.00|

|6|Komal |22| MP |4500.00|

+--+-----+----+-----+-----------+

The following program would update the table and increase salary of each customer by 500 and use the SQL%ROWCOUNT attribute to determine the number of rows affected:

DECLARE

total\_rows number(2);

BEGIN

UPDATE customers

SET salary = salary +500;

IF sql%notfound THEN

dbms\_output.put\_line('no customers selected');

ELSIF sql%found THEN

total\_rows:=sql%rowcount;

dbms\_output.put\_line(total\_rows||' customers selected ');

END IF;

END;

/

When the above code is executed at SQL prompt, it produces the following result:

6 customers selected

PL/SQL procedure successfully completed.

If you check the records in customers table, you will find that the rows have been updated:

Select\*from customers;

+----+-----+-----+--------+-------+

| ID | NAME| AGE | ADDRESS| SALARY|

+----+-----+-----+--------+-------+

|1|Ramesh|32|Ahmedabad|2500.00|

|2|Khilan|25|Delhi|2000.00|

|3|kaushik|23|Kota|2500.00|

|4|Chaitali|25|Mumbai|7000.00|

|5|Hardik|27|Bhopal|9000.00|

|6|Komal|22| MP |5000.00|

+----+-----+----+-----+-----------+

## Explicit Cursors

Explicit cursors are programmer defined cursors for gaining more control over the **context area**. An explicit cursor should be defined in the declaration section of the PL/SQL Block. It is created on a SELECT Statement which returns more than one row.

The syntax for creating an explicit cursor is:

CURSOR cursor\_name IS select\_statement;

Working with an explicit cursor involves four steps:

* Declaring the cursor for initializing in the memory
* Opening the cursor for allocating memory
* Fetching the cursor for retrieving data
* Closing the cursor to release allocated memory

## Declaring the Cursor

Declaring the cursor defines the cursor with a name and the associated SELECT statement. For example:

CURSOR c\_customers IS

SELECT id, name, address FROM customers;

## Opening the Cursor

Opening the cursor allocates memory for the cursor and makes it ready for fetching the rows returned by the SQL statement into it. For example, we will open above-defined cursor as follows:

OPEN c\_customers;

## Fetching the Cursor

Fetching the cursor involves accessing one row at a time. For example we will fetch rows from the above-opened cursor as follows:

FETCH c\_customers INTO c\_id,c\_name,c\_addr;

## Closing the Cursor

Closing the cursor means releasing the allocated memory. For example, we will close above-opened cursor as follows:

CLOSE c\_customers;

## Example:

Following is a complete example to illustrate the concepts of explicit cursors:

DECLARE

c\_id customers.id%type;

c\_name customers.name%type;

c\_addr customers.address%type;

CURSOR c\_customersis

SELECT id, name, address FROM customers;

BEGIN

OPEN c\_customers;

LOOP

FETCH c\_customers into c\_id, c\_name, c\_addr;

EXIT WHEN c\_customers%notfound;

dbms\_output.put\_line(c\_id||' '||c\_name||' '||c\_addr);

END LOOP;

CLOSE c\_customers;

END;

/

When the above code is executed at SQL prompt, it produces the following result:

1RameshAhmedabad

2KhilanDelhi

3kaushikKota

4ChaitaliMumbai

5HardikBhopal

6Komal MP

**Procedure & Functions**

1. Function is mainly used in the case where **it must return a value**. Whereas a procedure may or may not return a value or may return more than one value using the OUT parameter.Procedure can return multiple values **(max 1024)**
2. **Function can be called from SQL** statements whereas procedure cannot be called from the sql statements
3. **Functions are normally used for computations** whereas procedures are normally used for **executing business logic.**
4. **You can have DML** (insert,update, delete) statements in a function. But, you cannot call such a function in a SQL query.
5. **Stored Procedure: supports deferred name resolution.** Example while writing a stored procedure that uses table named tabl1 and tabl2 etc. but actually not exists in database is allowed only in during creation but runtime throws error Function won’t support deferred name resolution.
6. **Stored procedure returns always integer value** by default zero. whereas **function return type could be scalar or table or table values**
7. **Stored procedure is precompiled** execution plan whereas functions are not.
8. **A procedure may modify an object where a function can only return a value** The RETURN statement immediately completes the execution of a subprogram and returns control to the caller.

**Varchar &Varchar2**

1. **VARCHAR** is reserved by Oracle to support distinction between NULL and empty string in future, as ANSI standard prescribes. **VARCHAR2**does not distinguish **between** a NULL and empty string, and never will. If we declare datatype as **VARCHAR then it will occupy space for NULL** values, In case of VARCHAR2 datatype it will not occupy any space.
2. VARCHAR can store up to 2000 bytes of characters while VARCHAR2 can store up to 4000 bytes of characters.
3. Varchar is of ANSI SQL standard while Varchar2 is of Oracle standard.
4. VARCHAR was previously of fixed length and right padded with NULL spaces to fill up the total space.

**Types of join**

**Delete & Truncate**

The DELETE command is used to remove rows from a table. A **WHERE clause** can be used to only remove some rows. If no WHERE condition is specified, all rows will be removed. After performing a DELETE operation you need to **COMMIT or ROLLBACK** the transaction to make the change permanent or to undo it. Note that this operation will cause all **DELETE triggers** on the table to fire.

**TRUNCATE is a DDL** command whereas DELETE is a DML command.

**TRUCATE is faster**

The DROP command removes a table from the database. All the tables' rows, indexes and privileges will also be removed. No DML triggers will be fired. The operation cannot be rolled back.

**UK and PK**

1. Can be more than one unique key in one table
2. Unique key can have null values
3. PK creates a Clustered index and UK creates a Non Clustered Index.
4. Primary Key is used to identify a row (record) in a table whereas Unique-key is to prevent duplicate values in a column.
5. PK can be used as a FK in other table. In SQL Server, UK can be made FK in another table.

**Parent key, Surrogate Key, Candidate Key**

A **parent key** is either a primary key or a unique key in the parent table of a referential constraint. The values of a parent key determine the valid values of the foreign key in the constraint. You must create a unique index on a parent key.

**Candidate Key** – A Candidate Key can be any column or a combination of columns that can **qualify as unique key in database**. There can be multiple Candidate Keys in one table. Each Candidate Key can qualify as Primary Key.

**Surrogate Key** - It is just a unique identifier or number for each row that can be used for the primary key to the table. The only requirement for a surrogate primary key is that it is unique for each row in the table.

Data warehouses typically use a surrogate, (also known as artificial or identity key), **key for the dimension tables** primary keys.

Surrogate key is system generated artificial primary key values.

Surrogate keys are that which join dimension tables and fact table

**Bitmap index & Bitwise Index**

The bitmap index is generally for columns with lots of duplicate values (low cardinality)-M/F,

while b-tree indexes are best for high cardinality columns-PK

**Oracle b-tree index**

The oldest and most popular type of Oracle indexing is a standard b-tree index, which excels at servicing simple queries. The b-tree index was introduced in the earliest releases of Oracle and remains widely used with Oracle. B-tree indexes are used to avoid large sorting operations. For example, a SQL query requiring 10,000 rows to be presented in sorted order will often use a b-tree index to avoid the very large sort required to deliver the data to the end user.

Oracle offers several options when creating an index using the default b-tree structure. It allows you to index on multiple columns (concatenated indexes) to improve access speeds. Also, it allows for individual columns to be sorted in different orders. For example, we could create a b-tree index on a column called last\_name in ascending order and have a second column within the index that displays the salary column in descending order.

*create index    name\_salary\_idx  
on   person*

*(   last\_nameasc,  
   salary desc);*

While b-tree indexes are great for simple queries, they are not very good for the following situations:

* **Low-cardinality columns** - Columns with less than 200 distinct values do not have the selectivity required in order to benefit from standard b-tree index structures.
* **No support for SQL functions** - B-tree indexes are not able to support SQL queries using Oracle's built-in functions. Oracle provides a variety of built-in functions that allow SQL statements to query on a piece of an indexed column or on any one of a number of transformations against the indexed column.

Prior to the introduction of **Oracle function-based indexes**, the Oracle cost-based SQL optimizer had to perform time-consuming long-table, full-table scans due to these shortcomings. Consequently, it was no surprise when Oracle introduced more robust types of indexing structures.

HexaDecimalRowID

**Bitmapped indexes**

Oracle bitmap indexes are very different from standard b-tree indexes. In bitmap structures, a two-dimensional array is created with one column for every row in the table being indexed. Each column represents a distinct value within the bitmapped index. This two-dimensional array represents each value within the index multiplied by the number of rows in the table. At row retrieval time, Oracle decompresses the bitmap into the RAM data buffers so it can be rapidly scanned for matching values. These matching values are delivered to Oracle in the form of a Row-ID list, and these Row-ID values may directly access the required information.

The real benefit of bitmapped indexing occurs when one table includes multiple bitmapped indexes. Each individual column may have low cardinality. The creation of multiple bitmapped indexes provides a very powerful method for rapidly answering difficult SQL queries.

For example, assume there is a motor vehicle database with numerous low-cardinality columns such as car\_color, car\_make, car\_model, and car\_year. Each column contains less than 100 distinct values by themselves, and a b-tree index would be fairly useless in a database of 20 million vehicles. However, combining these indexes together in a query can provide blistering response times a lot faster than the traditional method of reading each one of the 20 million rows in the base table. For example, assume we wanted to find old blue Toyota Corollas manufactured in 1981.

*Select   license\_plat\_nbr  
from   vehicle  
and   make = 'toyota'  
and   year = 1981;*

Oracle uses a specialized optimizer method called a bitmapped index merge to service this query. In a bitmapped index merge, each Row-ID, or RID, list is built independently by using the bitmaps, and a special merge routine is used in order to compare the RID lists and find the intersecting values. Using this methodology, Oracle can provide subsecond response time when working against multiple low-cardinality columns (Figure 7.2).

**Clustered Index.**

* In clustered index the non-leaf level actually points to the actual data.
* This type of index reorders the physical order of the table and search based on the key values.
* Each table can have only one clustered index.

**NonClustered Index.**

* In Non-Clustered index the leaf nodes point to pointers (they are rowid’s) which then point to actual data.
* NonClustered Index does not alter the physical order of the table and maintains logical order of data.
* Each table can have 999 nonclustered indexes.

**Function Based Index**

**Clustered index & Non Clustered index**

**VArrays, PLSQL table, Nested Table**

**Pragma Autonomous Transactions**

**Snapshot**

**Global Temp Table**

**Inline View**

**Correlated Subqueries**

**Top N analysis**

**Analytical Function**

**TOAD 12.6.0.53**

**-- query: tables having some data**

SELECT t.name AS table\_name,

i.rows

FROM sys.tables AS t INNER JOIN

sys.sysindexes AS i ON t.object\_id = i.id AND i.indid < 2 and rows > 0

order by i.rows,t.name

--- Query : duplicate columns

select lower(trim(scode)) , count(\*) , min(hmy) from ASFORMUL

group by lower(trim(scode))

having count(\*) > 1

Select \*

From Listprop2

where Hproperty in(

Select Hproperty-- , Hproplist ,Count(\*)

from listprop2

Group By Hproperty, Hproplist

Having Count(\*) > 1 )

order by 1,2

delete from ASFORMUL

where hmy in(

select min(hmy) from ASFORMUL

group by lower(trim(scode))

having count(\*) > 1)

--query : duplicate row

Select \* from listprop2

where hproplist = 2850

and hproperty = 2852

and rownum =

(Select min(rownum)

from listprop2

where hproplist = 2850

and hproperty = 2852)

**Top 19 values in single column :**

SELECT TOP 19 row\_number() OVER (ORDER BY NAME) Col\_name

FROM sysobjects

Select \* from information\_schema.COLUMNS

where TABLE\_NAME = 'owner'

SELECT \* FROM INFORMATION\_SCHEMA.COLUMNS

WHERE COLUMN\_NAME LIKE '%CONTRACTED%'

SELECT \* FROM INFORMATION\_SCHEMA.COLUMNS

WHERE COLUMN\_NAME LIKE '%LEASETYPE%'

AND TABLE\_NAME LIKE '%TENANT%'

QUESTION 1:  What is database?

ANSWER: A database is a logically coherent collection of data with some inherent meaning, representing some aspect of real world and which is designed, built and populated with data for a specific purpose.

QUESTION 2: What is DBMS?

ANSWER:   
? Redundancy is controlled.   
? Unauthorized access is restricted.   
? Providing multiple user interfaces.   
? Enforcing integrity constraints.   
? Providing backup and recovery.

QUESTION 3:  What is a Database system?

ANSWER:  The database and DBMS software together is called as Database system.

QUESTION 4: Disadvantage in File Processing System?

ANSWER:   
? Data redundancy & inconsistency.   
? Difficult in accessing data.   
? Data isolation.   
? Data integrity.   
? Concurrent access is not possible.   
? Security Problems. .

QUESTION 5: Describe the three levels of data abstraction?

ANSWER:   
The are three levels of abstraction:

? Physical level: The lowest level of abstraction describes how data are stored.

? Logical level: The next higher level of abstraction, describes what data are stored in database and what relationship among those data.

? View level: The highest level of abstraction describes only part of entire database.

QUESTION 6: Define the "integrity rules"

ANSWER: There are two Integrity rules.

? Entity Integrity: States that ?Primary key cannot have NULL value?

? Referential Integrity: States that ?Foreign Key can be either a NULL value or should be Primary Key value of other relation.

QUESTION 7: What is extension and intension?

ANSWER:   
Extension -It is the number of tuples present in a table at any instance. This is time dependent.

Intension – It is a constant value that gives the name, structure of table and the constraints laid on it.

QUESTION 8: What is System R? What are its two major subsystems?

ANSWER:   
System R was designed and developed over a period of 1974-79 at IBM San Jose Research Center . It is a prototype and its purpose was to demonstrate that it is possible to build a Relational System that can be used in a real life environment to solve real life problems, with performance at least comparable to that of existing system.

Its two subsystems are

? Research Storage

? System Relational Data System.

QUESTION 10: How is the data structure of System R different from the relational structure?

ANSWER:   
Unlike Relational systems in System R

? Domains are not supported

? Enforcement of candidate key uniqueness is optional

? Enforcement of entity integrity is optional

? Referential integrity is not enforced

QUESTION 11: What is Data Independence?

ANSWER: Data independence means that ?the application is independent of the storage structure and access strategy of data

?. In other words, The ability to modify the schema definition in one level should not affect the schema definition in the next higher level.

Two types of Data Independence:   
? Physical Data Independence : Modification in physical level should not affect the logical level.   
? Logical Data Independence : Modification in logical level should affect the view level.

NOTE: Logical Data Independence is more difficult to achieve

QUESTION 12: What is a view? How it is related to data independence?

ANSWER: A view may be thought of as a virtual table, that is, a table that does not really exist in its own right but is instead derived from one or more underlying base table. In other words, there is no stored file that direct represents the view instead a definition of view is stored in data dictionary.   
Growth and restructuring of base tables is not reflected in views. Thus the view can insulate users from the effects of restructuring and growth in the database. Hence accounts for logical data independence. .

QUESTION 13: What is Data Model?

ANSWER: A collection of conceptual tools for describing data, data relationships data semantics and constraints.

QUESTION 14: What is E-R model?

ANSWER: This data model is based on real world that consists of basic objects called entities and of relationship among these objects. Entities are described in a database by a set of attributes.

QUESTION 15: What is Object Oriented model?

ANSWER: This model is based on collection of objects. An object contains values stored in instance variables within the object. An object also contains bodies of code that operate on the object. These bodies of code are called methods. Objects that contain same types of values and the same methods are grouped together into classes.

QUESTION 16: What is an Entity?

ANSWER:

An entity is a real-world thing which can be distinctly identified like a person, place or a concept. It is an object which is distinguishable from others. If we cannot distinguish it from others then it is an object but not an entity. An entity can be of two types:

Tangible Entity: Tangible Entities are those entities which exist in the real world physically. Example: Person, car, etc.

Intangible Entity: Intangible Entities are those entities which exist only logically and have no physical existence. Example: Bank Account, etc.

QUESTION 17: What is an Entity type?

ANSWER: The entity type is a collection of the entity having similar attributes.

It is a collection (set) of entities that have same attributes.

Strong Entity: Strong entity are those entity types which has a key attribute. The primary key helps in identifying each entity uniquely.

Weak Entity: An entity set may not have sufficient attributes to form a primary key, however it contains a partial key.

QUESTION 18: What is an Entity set?

ANSWER: Entity Set is a collection of entities of the same entity type.

QUESTION 19: What is an Extension of entity type?

ANSWER: The collections of entities of a particular entity type are grouped together into an entity set.

QUESTION 21: What is an attribute?

ANSWER: It is a particular property, which describes the entity.

QUESTION 22: What is a Relation Schema and a Relation?

ANSWER: refers to the meta-data that describes the structure of data within a certain domain. It is the blueprint of a database that outlines the way its structure organizes data into tables

A relation Schema denoted by R(A1, A2, ?, An) is made up of the relation name R and the list of attributes Ai that it contains. A relation is defined as a set of tuples. Let r be the relation which contains set tuples (t1, t2, t3, …, tn). Each tuple is an ordered list of n-values t=(v1,v2, …, vn).

QUESTION 23: What is degree of a Relation?

ANSWER: It is the number of attribute of its relation schema.

QUESTION 24: What is Relationship?

ANSWER:  It is an association among two or more entities.

QUESTION 25: What is Relationship set?

ANSWER: The collection (or set) of similar relationships.

QUESTION 26: What is Relationship type?

ANSWER: Relationship type defines a set of associations or a relationship set among a given set of entity types.

QUESTION 27: What is degree of Relationship type?

ANSWER: It is the number of entity type participating.

QUESTION 28: What is Data Storage – Definition Language?

ANSWER:  The storage structures and access methods used by database system are specified by a set of definition in a special type of DDL called data storage-definition language.

QUESTION 29: What is DML (Data Manipulation Language)?

ANSWER: This language that enable user to access or manipulate data as organized by appropriate data model.

? Procedural DML or Low level: DML requires a user to specify what data are needed and how to get those data.

? Non-Procedural DML or High level: DML requires a user to specify what data are needed without specifying how to get those data.

QUESTION 30: What is VDL (View Definition Language)?

ANSWER: It specifies user views and their mappings to the conceptual schema.

QUESTION 31: What is DML Compiler?

ANSWER: It translates DML statements in a query language into low-level instruction that the query evaluation engine can understand.

QUESTION 32: What is Query evaluation engine?

ANSWER: It executes low-level instruction generated by compiler.

QUESTION 33: What is DDL Interpreter?

ANSWER: It interprets DDL statements and record them in tables containing metadata.

QUESTION 34: What is Record-at-a-time?

ANSWER: The Low level or Procedural DML can specify and retrieve each record from a set of records. This retrieve of a record is said to be Record-at-a-time.

QUESTION 35: What is Set-at-a-time or Set-oriented?

ANSWER: The High level or Non-procedural DML can specify and retrieve many records in a single DML statement. This retrieve of a record is said to be Set-at-a-time or Set-oriented.

QUESTION 36: What is Relational Algebra?

ANSWER: It is procedural query language. It consists of a set of operations that take one or two relations as input and produce a new relation.

QUESTION 37: What is Relational Calculus?

ANSWER: It is an applied predicate calculus specifically tailored for relational databases proposed by E.F. Codd. E.g. of languages based on it are DSL ALPHA, QUEL.

QUESTION 38: How does Tuple-oriented relational calculus differ from domain-oriented relational calculus

ANSWER: The tuple-oriented calculus uses a tuple variables i.e., variable whose only permitted values are tuples of that relation. E.g. QUEL

The domain-oriented calculus has domain variables i.e., variables that range over the underlying domains instead of over relation. E.g. ILL, DEDUCE.

QUESTION 39: What is normalization?

ANSWER: It is a process of analyzing the given relation schemas based on their Functional Dependencies (FDs) and primary key to achieve the properties

? Minimizing redundancy

? Minimizing insertion, deletion and update anomalies.

QUESTION 40: What is Functional Dependency?

ANSWER: A Functional dependency is denoted by X Y between two sets of attributes X and Y that are subsets of R specifies a constraint on the possible tuple that can form a relation state r of R. The constraint is for any two tuples t1 and t2 in r if t1[X] = t2[X] then they have t1[Y] = t2[Y]. This means the value of X component of a tuple uniquely determines the value of component Y.

QUESTION 41: When is a functional dependency F said to be minimal?

ANSWER: ? Every dependency in F has a single attribute for its right hand side.

? We cannot replace any dependency X A in F with a dependency Y A where Y is a proper subset of X and still have a set of dependency that is equivalent to F.

? We cannot remove any dependency from F and still have set of dependency that is equivalent to F.

QUESTION 42: What is Multivalued dependency?

ANSWER:  Multivalued dependency denoted by X Y specified on relation schema R, where X and Y are both subsets of R, specifies the following constraint on any relation r of R: if two tuples t1 and t2 exist in r such that t1[X] = t2[X] then t3 and t4 should also exist in r with the following properties

? t3[x] = t4[X] = t1[X] = t2[X]

? t3[Y] = t1[Y] and t4[Y] = t2[Y]

? t3[Z] = t2[Z] and t4[Z] = t1[Z]

where [Z = (R-(X U Y)) ]

QUESTION 43: What is Lossless join property?

ANSWER: It guarantees that the spurious tuple generation does not occur with respect to relation schemas after decomposition.

QUESTION 44: What is 1 NF (Normal Form)?

ANSWER: The domain of attribute must include only atomic (simple, indivisible) values.

QUESTION 45: What is Fully Functional dependency?

ANSWER: It is based on concept of full functional dependency. A functional dependency X Y is full functional dependency if removal of any attribute A from X means that the dependency does not hold any more.

QUESTION 46: What is 2NF?

ANSWER: A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on primary key.

QUESTION 47: What is 3NF?

ANSWER: A relation schema R is in 3NF if it is in 2NF and for every FD X A either of the following is true

? X is a Super-key of R.

? A is a prime attribute of R.

In other words, if every non prime attribute is non-transitively dependent on primary key.

QUESTION 48: What is BCNF (Boyce-Codd Normal Form)?

ANSWER: A relation schema R is in BCNF if it is in 3NF and satisfies an additional constraint that for every FD X A, X must be a candidate key.

QUESTION 49: What is 4NF?

ANSWER: A relation schema R is said to be in 4NF if for every Multivalued dependency X Y that holds over R, one of following is true

? X is subset or equal to (or) XY = R.

? X is a super key.

QUESTION 50: What is 5NF?

ANSWER: A Relation schema R is said to be 5NF if for every join dependency {R1, R2, …, Rn} that holds R, one the following is true

? Ri = R for some i.

? The join dependency is implied by the set of FD, over R in which the left side is key of R.

5. What is the parameter substitution symbol used with INSERT INTO command?  
&

6. Which command displays the SQL command in the SQL buffer, and then executes it?  
RUN

7. What are the wildcards used for pattern matching?  
\_ for single character substitution and % for multi-character substitution

8. State true or false. EXISTS, SOME, ANY are operators in SQL.  
True

9. State true or false. !=, <>, ^= all denote the same operation.  
True

10. What are the privileges that can be granted on a table by a user to others?  
Insert, update, delete, select, references, index, execute, alter, all

11. What command is used to get back the privileges offered by the GRANT command?  
REVOKE

12. Which system tables contain information on privileges granted and privileges obtained?  
USER\_TAB\_PRIVS\_MADE, USER\_TAB\_PRIVS\_RECD

13. Which system table contains information on constraints on all the tables created?  
USER\_CONSTRAINTS

14. TRUNCATE TABLE EMP;  
DELETE FROM EMP;  
Will the outputs of the above two commands differ?  
Both will result in deleting all the rows in the table EMP.

15. What is the difference between TRUNCATE and DELETE commands?  
TRUNCATE is a DDL command whereas DELETE is a DML command. Hence DELETE operation can be rolled back, but TRUNCATE operation cannot be rolled back. WHERE clause can be used with DELETE and not with TRUNCATE.

16. What command is used to create a table by copying the structure of another table?  
Answer :  
CREATE TABLE .. AS SELECT command  
Explanation :  
To copy only the structure, the WHERE clause of the SELECT command should contain a FALSE statement as in the following.  
CREATE TABLE NEWTABLE AS SELECT \* FROM EXISTINGTABLE WHERE 1=2;  
If the WHERE condition is true, then all the rows or rows satisfying the condition will be copied to the new table.

What is RDBMS?  
Relational Data Base Management Systems (RDBMS) are database management systems that maintain data records and indices in tables. Relationships may be created and maintained across and among the data and tables. In a relational database, relationships between data items are expressed by means of tables. Interdependencies among these tables are expressed by data values rather than by pointers. This allows a high degree of data independence. An RDBMS has the capability to recombine the data items from different files, providing powerful tools for data usage.  
  
What is normalization?  
Database normalization is a data design and organization process applied to data structures based on rules that help build relational databases. In relational database design, the process of organizing data to minimize redundancy. Normalization usually involves dividing a database into two or more tables and defining relationships between the tables. The objective is to isolate data so that additions, deletions, and modifications of a field can be made in just one table and then propagated through the rest of the database via the defined relationships.  
  
What are different normalization forms?  
1NF: Eliminate Repeating Groups  
Make a separate table for each set of related attributes, and give each table a primary key. Each field contains at most one value from its attribute domain.  
  
2NF: Eliminate Redundant Data  
If an attribute depends on only part of a multi-valued key, remove it to a separate table.  
  
3NF: Eliminate Columns Not Dependent On Key  
If attributes do not contribute to a description of the key, remove them to a separate table. All attributes must be directly dependent on the primary key  
  
BCNF: Boyce-Codd Normal Form  
If there are non-trivial dependencies between candidate key attributes, separate them out into distinct tables.  
  
4NF: Isolate Independent Multiple Relationships  
No table may contain two or more 1:n or n:m relationships that are not directly related.  
  
5NF: Isolate Semantically Related Multiple Relationships  
There may be practical constrains on information that justify separating logically related many-to-many relationships.  
  
ONF: Optimal Normal Form  
A model limited to only simple (elemental) facts, as expressed in Object Role Model notation.  
  
DKNF: Domain-Key Normal Form  
A model free from all modification anomalies. Remember, these normalization guidelines are cumulative.

For a database to be in 3NF, it must first fulfill all the criteria of a 2NF and 1NF database.  
  
What is Stored Procedure?  
A stored procedure is a named group of SQL statements that have been previously created and stored in the server database. Stored procedures accept input parameters so that a single procedure can be used over the network by several clients using different input data. And when the procedure is modified, all clients automatically get the new version. Stored procedures reduce network traffic and improve performance. Stored procedures can be used to help ensure the integrity of the database.  
e.g. sp\_helpdb, sp\_renamedb, sp\_depends etc.  
  
Read more: <http://www.placementpapers.us/general/386-sql_interview_questions_answers.html#ixzz1dpvQ9Ol8>   
Under Creative Commons License: [Attribution](http://creativecommons.org/licenses/by/3.0)

Database (DBMS) interview questions and answers are below

Questions : 1 What is database or database management systems (DBMS)? and - What’s the difference between file and database? Can files qualify as a database?

Answers : 1 Database provides a systematic and organized way of storing, managing and retrieving from collection of logically related information.

Secondly the information has to be persistent, that means even after the application is closed the information should be persisted.

Finally it should provide an independent way of accessing data and should not be dependent on the application to access the information.

Main difference between a simple file and database that database has independent way (SQL) of accessing information while simple files do not File meets the storing, managing and retrieving part of a database but not the independent way of accessing data. Many experienced programmers think that the main difference is that file can not provide multi-user capabilities which a DBMS provides. But if we look at some old COBOL and C programs where file where the only means of storing data, we can see functionalities like locking, multi-user etc provided very efficiently. So it’s a matter of debate if some interviewers think this as a main difference between files and database accept it… going in to debate is probably loosing a job.

Questions : 2 What is SQL ?

Answers : 2 SQL stands for Structured Query Language.SQL is an ANSI (American National Standards Institute) standard computer language for accessing and manipulating database systems. SQL statements are used to retrieve and update data in a database.

Questions : 3 What’s difference between DBMS and RDBMS ?

Answers : 3 DBMS provides a systematic and organized way of storing, managing and retrieving from collection of logically related information. RDBMS also provides what DBMS provides but above that it provides relationship integrity. So in short we can say

RDBMS = DBMS + REFERENTIAL INTEGRITY

These relations are defined by using “Foreign Keys” in any RDBMS.Many DBMS companies claimed there DBMS product was a RDBMS compliant, but according to industry rules and regulations if the DBMS fulfills the twelve CODD rules it’s truly a RDBMS. Almost all DBMS (SQL SERVER, ORACLE etc) fulfills all the twelve CODD rules and are considered as truly RDBMS.

Questions : 4 What are CODD rules?

Answers : 4 In 1969 Dr. E. F. Codd laid down some 12 rules which a DBMS should adhere in order to get the logo of a true RDBMS.

Rule 1: Information Rule.

"All information in a relational data base is represented explicitly at the logical level and in exactly one way - by values in tables."

Rule 2: Guaranteed access Rule.

"Each and every datum (atomic value) in a relational data base is guaranteed to be logically accessible by resorting to a combination of table name, primary key value and column name."

In flat files we have to parse and know exact location of field values. But if a DBMS is truly RDBMS you can access the value by specifying the table name, field name, for instance Customers.Fields [‘Customer Name’].

Rule 3: Systematic treatment of null values.

"Null values (distinct from the empty character string or a string of blank characters and distinct from zero or any other number) are supported in fully relational DBMS for representing missing information and inapplicable information in a systematic way, independent of data type.".

Rule 4: Dynamic on-line catalog based on the relational model.

"The data base description is represented at the logical level in the same way as ordinary data, so that authorized users can apply the same relational language to its interrogation as they apply to the regular data."The Data Dictionary is held within the RDBMS, thus there is no-need for off-line volumes to tell you the structure of the database.

Rule 5: Comprehensive data sub-language Rule.

"A relational system may support several languages and various modes of terminal use (for example, the fill-in-the-blanks mode). However, there must be at least one language whose statements are expressible, per some well-defined syntax, as character strings and that is comprehensive in supporting all the following items

Data Definition

View Definition

Data Manipulation (Interactive and by program).

Integrity Constraints

Authorization.

Transaction boundaries ( Begin , commit and rollback)

Rule 6: .View updating Rule

"All views that are theoretically updatable are also updatable by the system."

Rule 7: High-level insert, update and delete.

"The capability of handling a base relation or a derived relation as a single operand applies not only to the retrieval of data but also to the insertion, update and deletion of data."

Rule 8: Physical data independence.

"Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representations or access methods."

Rule 9: Logical data independence.

"Application programs and terminal activities remain logically unimpaired when information-preserving changes of any kind that theoretically permit un-impairment are made to the base tables."

Rule 10: Integrity independence.

"Integrity constraints specific to a particular relational data base must be definable in the relational data sub-language and storable in the catalog, not in the application programs." Rule 11: Distribution independence.

"A relational DBMS has distribution independence."

Rule 12: Non-subversion Rule.

"If a relational system has a low-level (single-record-at-a-time) language, that low level cannot be used to subvert or bypass the integrity Rules and constraints expressed in the higher level relational language (multiple-records-at-a-time)."

Questions : 5 What are E-R diagrams?

Answers : 5 E-R diagram also termed as Entity-Relationship diagram shows relationship between various tables in the database. .

Questions : 6 How many types of relationship exist in database designing?

Answers : 6 There are three major relationship models:-

One-to-one

One-to-many

Many-to-many

Questions : 7 7.What is normalization? What are different type of normalization?

Answers : 7 There is set of rules that has been established to aid in the design of tables that are meant to be connected through relationships. This set of rules is known as Normalization.

Benefits of Normalizing your database include:

=>Avoiding repetitive entries

=>Reducing required storage space

=>Preventing the need to restructure existing tables to accommodate new data.

=>Increased speed and flexibility of queries, sorts, and summaries.

Following are the three normal forms :-

First Normal Form

For a table to be in first normal form, data must be broken up into the smallest un possible.In addition to breaking data up into the smallest meaningful values, tables first normal form should not contain repetitions groups of fields.

Second Normal form

The second normal form states that each field in a multiple field primary keytable must be directly related to the entire primary key. Or in other words,each non-key field should be a fact about all the fields in the primary key.

Third normal form

A non-key field should not depend on other Non-key field.

Questions : 8 What is denormalization ?

Answers : 8 Denormalization is the process of putting one fact in numerous places (its vice-versa of normalization).Only one valid reason exists for denormalizing a relational design - to enhance performance.The sacrifice to performance is that you increase redundancy in database.

Questions : 9 Can you explain Fourth Normal Form and Fifth Normal Form ?

Answers : 9 In fourth normal form it should not contain two or more independent multi-v about an entity and it should satisfy “Third Normal form”.

Fifth normal form deals with reconstructing information from smaller pieces of information. These smaller pieces of information can be maintained with less redundancy.

Questions : 10 Have you heard about sixth normal form?

Answers : 10 If we want relational system in conjunction with time we use sixth normal form. At this moment SQL Server does not supports it directly.

Questions : 11 What are DML and DDL statements?

Answers : 11 DML stands for Data Manipulation Statements. They update data values in table. Below are the most important DDL statements:-

=>SELECT - gets data from a database table

=> UPDATE - updates data in a table

=> DELETE - deletes data from a database table

=> INSERT INTO - inserts new data into a database table

DDL stands for Data definition Language. They change structure of the database objects like table, index etc. Most important DDL statements are as shown below:-

=>CREATE TABLE - creates a new table in the database.

=>ALTER TABLE – changes table structure in database.

=>DROP TABLE - deletes a table from database

=> CREATE INDEX - creates an index

=> DROP INDEX - deletes an index

Questions : 12 How do we select distinct values from a table?

Answers : 12 DISTINCT keyword is used to return only distinct values. Below is syntax:- Column age and Table pcdsEmp

SELECT DISTINCT age FROM pcdsEmp

Questions : 13 What is Like operator for and what are wild cards?

Answers : 13 LIKE operator is used to match patterns. A "%" sign is used to define the pattern.

Below SQL statement will return all words with letter "S"

SELECT \* FROM pcdsEmployee WHERE EmpName LIKE 'S%'

Below SQL statement will return all words which end with letter "S"

SELECT \* FROM pcdsEmployee WHERE EmpName LIKE '%S'

Below SQL statement will return all words having letter "S" in between

SELECT \* FROM pcdsEmployee WHERE EmpName LIKE '%S%'

"\_" operator (we can read as “Underscore Operator”). “\_” operator is the character defined at that point. In the below sample fired a query Select name from pcdsEmployee where name like '\_s%' So all name where second letter is “s” is returned.

Questions : 14 Can you explain Insert, Update and Delete query?

Answers : 14 Insert statement is used to insert new rows in to table. Update to update existing data in the table. Delete statement to delete a record from the table. Below code snippet for Insert, Update and Delete :-

INSERT INTO pcdsEmployee SET name='rohit',age='24';

UPDATE pcdsEmployee SET age='25' where name='rohit';

DELETE FROM pcdsEmployee WHERE name = 'sonia';

Questions : 15 What is order by clause?

Answers : 15 ORDER BY clause helps to sort the data in either ascending order to descending order.

Ascending order sort query

SELECT name,age FROM pcdsEmployee ORDER BY age ASC

Descending order sort query

SELECT name FROM pcdsEmployee ORDER BY age DESC

Questions : 16 What is the SQL " IN " clause?

Answers : 16 SQL IN operator is used to see if the value exists in a group of values. For instance the below SQL checks if the Name is either 'rohit' or 'Anuradha' SELECT \* FROM pcdsEmployee WHERE name IN ('Rohit','Anuradha') Also you can specify a not clause with the same. SELECT \* FROM pcdsEmployee WHERE age NOT IN (17,16)

Questions : 17 Can you explain the between clause?

Answers : 17 Below SQL selects employees born between '01/01/1975' AND '01/01/1978' as per mysql

SELECT \* FROM pcdsEmployee WHERE DOB BETWEEN '1975-01-01' AND '2011-09-28'

Questions : 18 we have an employee salary table how do we find the second highest from it?

Answers : 18 below Sql Query find the second highest salary

SELECT \* FROM EmpSalary a

WHERE (2=(SELECT COUNT(DISTINCT(b.salary))

FROM EmpSalary b

WHERE b.salary>=a.salary))

Questions : 19 What are different types of joins in SQL?

Answers : 19 INNER JOIN

Inner join shows matches only when they exist in both tables. Example in the below SQL there are two tables Customers and Orders and the inner join in made on Customers.Customerid and Orders.Customerid. So this SQL will only give you result with customers who have orders. If the customer does not have order it will not display that record.

SELECT Customers.\*, Orders.\* FROM Customers INNER JOIN Orders ON Customers.CustomerID =Orders.CustomerID

LEFT OUTER JOIN

Left join will display all records in left table of the SQL statement. In SQL below customers with or without orders will be displayed. Order data for customers without orders appears as NULL values. For example, you want to determine the amount ordered by each customer and you need to see who has not ordered anything as well. You can also see the LEFT OUTER JOIN as a mirror image of the RIGHT OUTER JOIN (Is covered in the next section) if you switch the side of each table.

SELECT Customers.\*, Orders.\* FROM Customers LEFT OUTER JOIN Orders ON Customers.CustomerID =Orders.CustomerID

RIGHT OUTER JOIN

Right join will display all records in right table of the SQL statement. In SQL below all orders with or without matching customer records will be displayed. Customer data for orders without customers appears as NULL values. For example, you want to determine if there are any orders in the data with undefined CustomerID values (say, after a conversion or something like it). You can also see the RIGHT OUTER JOIN as a mirror image of the LEFT OUTER JOIN if you switch the side of each table.

SELECT Customers.\*, Orders.\* FROM Customers RIGHT OUTER JOIN Orders ON Customers.CustomerID =Orders.CustomerID

Questions : 20 What is “CROSS JOIN”? or What is Cartesian product?

Answers : 20 “CROSS JOIN” or “CARTESIAN PRODUCT” combines all rows from both tables. Number of rows will be product of the number of rows in each table. In real life scenario I can not imagine where we will want to use a Cartesian product. But there are scenarios where we would like permutation and combination probably Cartesian would be the easiest way to achieve it.

Questions : 21 How to select the first record in a given set of rows?

Answers : 21 Select top 1 \* from sales.salesperson

Questions : 22 What is the default “-SORT ” order for a SQL?

Answers : 22 ASCENDING

Questions : 23 What is a self-join?

Answers : 23 If we want to join two instances of the same table we can use self-join.

Questions : 24 What’s the difference between DELETE and TRUNCATE ?

Answers : 24 Following are difference between them:

=>>DELETE TABLE syntax logs the deletes thus making the delete operations low. TRUNCATE table does not log any information but it logs information about deallocation of data page of the table. So TRUNCATE table is faster as compared to delete table.

=>>DELETE table can have criteria while TRUNCATE can not.

=>> TRUNCATE table can not have triggers.

Questions : 25 What’s the difference between “UNION” and “UNION ALL” ?

Answers : 25 UNION SQL syntax is used to select information from two tables. But it selects only distinct records from both the table. , while UNION ALL selects all records from both the tables.

Questions : 26 What are cursors and what are the situations you will use them?

Answers : 26 SQL statements are good for set at a time operation. So it is good at handling set of data. But there are scenarios where we want to update row depending on certain criteria. we will loop through all rows and update data accordingly. There’s where cursors come in to picture.

Questions : 27 What is " Group by " clause?

Answers : 27 “Group by” clause group similar data so that aggregate values can be derived.

Questions : 28 What is the difference between “HAVING” and “WHERE” clause?

Answers : 28 “HAVING” clause is used to specify filtering criteria for “GROUP BY”, while “WHERE” clause applies on normal SQL.

Questions : 29 What is a Sub-Query?

Answers : 29 A query nested inside a SELECT statement is known as a subquery and is an alternative to complex join statements. A subquery combines data from multiple tables and returns results that are inserted into the WHERE condition of the main query. A subquery is always enclosed within parentheses and returns a column. A subquery can also be referred to as an inner query and the main query as an outer query. JOIN gives better performance than a subquery when you have to check for the existence of records.

For example, to retrieve all EmployeeID and CustomerID records from the ORDERS table that have the EmployeeID greater than the average of the EmployeeID field, you can create a nested query, as shown:

SELECT DISTINCT EmployeeID, CustomerID FROM ORDERS WHERE EmployeeID > (SELECT AVG(EmployeeID) FROM ORDERS)

Questions : 30 What are Aggregate and Scalar Functions?

Answers : 30 Aggregate and Scalar functions are in built function for counting and calculations.

Aggregate functions operate against a group of values but returns only one value.

AVG(column) :- Returns the average value of a column

COUNT(column) :- Returns the number of rows (without a NULL value) of a column

COUNT(\*) :- Returns the number of selected rows

MAX(column) :- Returns the highest value of a column

MIN(column) :- Returns the lowest value of a column

Scalar functions operate against a single value and return value on basis of the single value.

UCASE(c) :- Converts a field to upper case

LCASE(c) :- Converts a field to lower case

MID(c,start[,end]) :- Extract characters from a text field

LEN(c) :- Returns the length of a text

Questions : 31 Can you explain the SELECT INTO Statement?

Answers : 31 SELECT INTO statement is used mostly to create backups. The below SQL backsup the Employee table in to the EmployeeBackUp table. One point to be noted is that the structure of pcdsEmployeeBackup and pcdsEmployee table should be same. SELECT \* INTO pcdsEmployeeBackup FROM pcdsEmployee

Questions : 32 What is a View?

Answers : 32 View is a virtual table which is created on the basis of the result set returned by the select statement.

CREATE VIEW [MyView] AS SELECT \* from pcdsEmployee where LastName = 'singh'

In order to query the view

SELECT \* FROM [MyView]

Questions : 33 What is SQl injection ?

Answers : 33 It is a Form of attack on a database-driven Web site in which the attacker executes unauthorized SQL commands by taking advantage of insecure code on a system connected to the Internet, bypassing the firewall. SQL injection attacks are used to steal information from a database from which the data would normally not be available and/or to gain access to an organization’s host computers through the computer that is hosting the database.

SQL injection attacks typically are easy to avoid by ensuring that a system has strong input validation.

As name suggest we inject SQL which can be relatively dangerous for the database. Example this is a simple SQL

SELECT email, passwd, login\_id, full\_name

FROM members WHERE email = 'x'

Now somebody does not put “x” as the input but puts “x ; DROP TABLE members;”.

So the actual SQL which will execute is :-

SELECT email, passwd, login\_id, full\_name FROM members WHERE email = 'x' ; DROP TABLE members;

Think what will happen to your database.

Questions : 34 What is Data Warehousing ?

Answers : 34 Data Warehousing is a process in which the data is stored and accessed from central location and is meant to support some strategic decisions. Data Warehousing is not a requirement for Data mining. But just makes your Data mining process more efficient.

Data warehouse is a collection of integrated, subject-oriented databases designed to support the decision-support functions (DSF), where each unit of data is relevant to some moment in time.

Questions : 35 What are Data Marts?

Answers : 35 Data Marts are smaller section of Data Warehouses. They help data warehouses collect data. For example your company has lot of branches which are spanned across the globe. Head-office of the company decides to collect data from all these branches for anticipating market. So to achieve this IT department can setup data mart in all branch offices and a central data warehouse where all data will finally reside.

Questions : 36 What are Fact tables and Dimension Tables ? What is Dimensional Modeling and Star Schema Design

Answers : 36 When we design transactional database we always think in terms of normalizing design to its least form. But when it comes to designing for Data warehouse we think more in terms of denormalizing the database. Data warehousing databases are designed using Dimensional Modeling. Dimensional Modeling uses the existing relational database structure and builds on that.

There are two basic tables in dimensional modeling:-

Fact Tables.

Dimension Tables.

Fact tables are central tables in data warehousing. Fact tables have the actual aggregate values which will be needed in a business process. While dimension tables revolve around fact tables. They describe the attributes of the fact tables.

Questions : 37 What is Snow Flake Schema design in database? What’s the difference between Star and Snow flake schema?

Answers : 37 Star schema is good when you do not have big tables in data warehousing. But when tables start becoming really huge it is better to denormalize. When you denormalize star schema it is nothing but snow flake design. For instance below customeraddress table is been normalized and is a child table of Customer table. Same holds true for Salesperson table.

Questions : 38 What is ETL process in Data warehousing? What are the different stages in “Data warehousing”?

Answers : 38 ETL (Extraction, Transformation and Loading) are different stages in Data warehousing. Like when we do software development we follow different stages like requirement gathering, designing, coding and testing. In the similar fashion we have for data warehousing.

Extraction:-

In this process we extract data from the source. In actual scenarios data source can be in many forms EXCEL, ACCESS, Delimited text, CSV (Comma Separated Files) etc. So extraction process handle’s the complexity of understanding the data source and loading it in a structure of data warehouse.

Transformation:-

This process can also be called as cleaning up process. It’s not necessary that after the extraction process data is clean and valid. For instance all the financial figures have NULL values but you want it to be ZERO for better analysis. So you can have some kind of stored procedure which runs through all extracted records and sets the value to zero.

Loading:-

After transformation you are ready to load the information in to your final data warehouse database.

Questions : 39 What is Data mining ?

Answers : 39 Data mining is a concept by which we can analyze the current data from different perspectives and summarize the information in more useful manner. It’s mostly used either to derive some valuable information from the existing data or to predict sales to increase customer market.

There are two basic aims of Data mining:-

Prediction: -

From the given data we can focus on how the customer or market will perform. For instance we are having a sale of 40000 $ per month in India, if the same product is to be sold with a discount how much sales can the company expect.

Summarization: -

To derive important information to analyze the current business scenario. For example a weekly sales report will give a picture to the top management how we are performing on a weekly basis?

Questions : 40 Compare Data mining and Data Warehousing ?

Answers : 40 “Data Warehousing” is technical process where we are making our data centralized while “Data mining” is more of business activity which will analyze how good your business is doing or predict how it will do in the future coming times using the current data. As said before “Data Warehousing” is not a need for “Data mining”. It’s good if you are doing “Data mining” on a “Data Warehouse” rather than on an actual production database. “Data Warehousing” is essential when we want to consolidate data from different sources, so it’s like a cleaner and matured data which sits in between the various data sources and brings then in to one format. “Data Warehouses” are normally physical entities which are meant to improve accuracy of “Data mining” process. For example you have 10 companies sending data in different format, so you create one physical database for consolidating all the data from different company sources, while “Data mining” can be a physical model or logical model. You can create a database in “Data mining” which gives you reports of net sales for this year for all companies. This need not be a physical database as such but a simple query.

Questions : 41 What are indexes? What are B-Trees?

Answers : 41 Index makes your search faster. So defining indexes to your database will make your search faster.Most of the indexing fundamentals use “B-Tree” or “Balanced-Tree” principle. It’s not a principle that is something is created by SQL Server or ORACLE but is a mathematical derived fundamental.In order that “B-tree” fundamental work properly both of the sides should be balanced.

Questions : 42 I have a table which has lot of inserts, is it a good database design to create indexes on that table?

Insert’s are slower on tables which have indexes, justify it?or Why do page splitting happen?

Answers : 42 All indexing fundamentals in database use “B-tree” fundamental. Now whenever there is new data inserted or deleted the tree tries to become unbalance.

Creates a new page to balance the tree.

Shuffle and move the data to pages.

So if your table is having heavy inserts that means it’s transactional, then you can visualize the amount of splits it will be doing. This will not only increase insert time but will also upset the end-user who is sitting on the screen. So when you forecast that a table has lot of inserts it’s not a good idea to create indexes.

Questions : 43 What are the two types of indexes and explain them in detail? or What’s the difference between clustered and non-clustered indexes?

Answers : 43 There are basically two types of indexes:-

Clustered Indexes.

Non-Clustered Indexes.

In clustered index the non-leaf level actually points to the actual data.In Non-Clustered index the leaf nodes point to pointers (they are rowid’s) which then point to actual data.

**SQL interview questions and answers**

1. **What are two methods of retrieving SQL?**

2. **What cursor type do you use to retrieve multiple recordsets?**

3. **What is the difference between a "where" clause and a "having" clause?**

- "Where" is a kind of restriction statement. You use where clause to restrict all the data from DB. Where clause is using before result retrieving.

- But Having clause is using after retrieving the data. Having clause is a kind of filtering command.

4. **What is the basic form of a SQL statement to read data out of a table?**

The basic form to read data out of table is ‘SELECT \* FROM table\_name; ‘ An answer: ‘SELECT

\* FROM table\_name WHERE xyz= ‘whatever’;’ cannot be called basic form because of WHERE clause.

5. **What structure can you implement for the database to speed up table reads?** -

Follow the rules of DB tuning we have to: 1] properly use indexes ( different types of indexes) 2] properly locate different DB objects across different tablespaces, files and so on.3] create a special space (tablespace) to locate some of the data with special datatype (for example CLOB, LOB and …)

6. **What are the tradeoffs with having indexes**? -

1. Faster selects, slower updates. 2.Extra storage space to store indexes. Updates are slower because in addition to updating the table you have to update the index.

7. **What is a "join"?** - ‘join’ used to connect two or more tables logically with or without common field.

8. **What is "normalization"? "Denormalization"? Why do you sometimes want to denormalize?** - Normalizing data means eliminating redundant information from a table and organizing the data so that future changes to the table are easier. Denormalization means allowing redundancy in a table. The main benefit of denormalization is improved performance with simplified data retrieval and manipulation. This is done by reduction in the number of joins needed for data processing.

9. **What is a "constraint"?** - A constraint allows you to apply simple referential integrity checks to a table. There are four primary types of constraints that are currently supported by SQL Server: PRIMARY/UNIQUE - enforces uniqueness of a particular table column.

DEFAULT - specifies a default value for a column in case an insert operation does not provide one. FOREIGN KEY - validates that every value in a column exists in a column of another table.

CHECK - checks that every value stored in a column is in some specified list.

Each type of constraint performs a specific type of action. Default is not a constraint.

NOT NULL is one more constraint which does not allow values in the specific column to be null. And also it the only constraint which is not a table level constraint.

10. **What types of index data structures can you have?**

- An index helps to faster search values in tables. The three most commonly used index-types are:

- B-Tree: builds a tree of possible values with a list of row IDs that have the leaf value. Needs a lot of space and is the default index type for most databases.

- Bitmap: string of bits for each possible value of the column. Each bit string has one bit for each row. Needs only few space and is very fast.(however, domain of value cannot be large, e.g. SEX(m,f);

degree(BS,MS,PHD)

- Hash: A hashing algorithm is used to assign a set of characters to represent a text string such as a composite of keys or partial keys, and compresses the underlying data. Takes longer to build and is supported by relatively few databases.

11. **What is a "primary key"?**

- A PRIMARY INDEX or PRIMARY KEY is something which comes mainly from database theory. From its behavior is almost the same as an UNIQUE INDEX, i.e. there may only be one of each value in this column. If you call such an INDEX PRIMARY instead of UNIQUE, you say something about your table design, which I am not able to explain in few words. Primary Key is a type of a constraint enforcing uniqueness and data integrity for each row of a table. All columns participating in a primary key constraint must possess the NOT NULL property.

12. **What is a "functional dependency"? How does it relate to database table design**? -

Functional dependency relates to how one object depends upon the other in the database. for example, procedure/function sp2 may be called by procedure sp1. Then we say that sp1 has functional dependency on sp2.

13. **What is a "trigger"?**

- Triggers are stored procedures created in order to enforce integrity rules in a database. A trigger is executed every time a data-modification operation occurs (i.e., insert, update or delete). Triggers are executed automatically on occurrence of one of the data-modification operations. A trigger is a database object directly associated with a particular table. It fires whenever a specific statement/type of statement is issued against that table. The types of statements are insert,update,delete and query statements. Basically, trigger is a set of SQL statements A trigger is a solution to the restrictions of a constraint. For instance: 1.A database column cannot carry PSEUDO columns as criteria where a trigger can. 2. A database constraint cannot refer old and new values for a row where a trigger can.

14. **Why can a "group by" or "order by" clause be expensive to process**?

- Processing of "group by" or "order by" clause often requires creation of Temporary tables to process

the results of the query. Which depending of the result set can be very expensive.

15. **What is "index covering" of a query**?

- Index covering means that "Data can be found only using indexes, without touching the tables"

16. **What types of join algorithms can you have?**

17. **What is a SQL view**? - An output of a query can be stored as a view. View acts like small table which meets our criterion. View is a precomplied SQL query which is used to select data from one or more tables. A view is like a table but it doesn’t physically take any space. View is a good way to present data in a particular format if you use that query quite often. View can also be used to restrict users from accessing the tables directly.

SQL

1.    How do you read transaction logs?   
2.    How do you reset or reseed the IDENTITY column?   
3.    How do you persist objects, permissions in tempdb?   
4.    How do you simulate a deadlock for testing purposes?   
5.    How do you rename an SQL Server computer?   
6.    How do you run jobs from T-SQL?   
7.    How do you restore single tables from backup in SQL Server 7.0/2000? In SQL Server 6.5?   
8.    Where to get the latest MDAC from?   
9.    I forgot/lost the sa password. What do I do?   
10.    I have only the .mdf file backup and no SQL Server database backups. Can I get my database back into SQL Server?   
11.    How do you add a new column at a specific position (say at the beginning of the table or after the second column) using ALTER TABLE command?   
12.    How do you change or alter a user defined data type?   
13.    How do you rename an SQL Server 2000 instance?   
14.    How do you capture/redirect detailed deadlock information into the error logs?   
15.    How do you remotely administer SQL Server?   
16.    What are the effects of switching SQL Server from ?Mixed mode? to ?Windows only? authentication mode? What are the steps required, to not break existing applications?   
17.    Is there a command to list all the tables and their associated filegroups?   
18.    How do you ship the stored procedures, user defined functions (UDFs), triggers, views of my application, in an encrypted form to my clients/customers? How do you protect intellectual property?   
19.    How do you archive data from my tables? Is there a built-in command or tool for this?   
20.    How do you troubleshoot ODBC timeout expired errors experienced by applications accessing SQL Server databases?   
21.    How do you restart SQL Server service automatically at regular intervals?   
22.    What is the T-SQL equivalent of IIF (immediate if/ternary operator) function of other programming languages?   
23.    How do you programmatically find out when the SQL Server service started?   
24.    How do you get rid of the time part from the date returned by GETDATE function?   
25.    How do you upload images or binary files into SQL Server tables?   
26.    How do you run an SQL script file that is located on the disk, using T-SQL?   
27.    How do you get the complete error message from T-SQL while error handling?   
28.    How do you get the first day of the week, last day of the week and last day of the month using T-SQL date functions?   
29.    How do you pass a table name, column name etc. to the stored procedure so that I can dynamically select from a table?   
30.    Error inside a stored procedure is not being raised to my front-end applications using ADO. But I get the error when I run the procedure from Query Analyzer.   
31.    How do you suppress error messages in stored procedures/triggers etc. using T-SQL?   
32.    How do you save the output of a query/stored procedure to a text file?   
33.    How do you join tables from different databases?   
34.    How do you join tables from different servers?   
35.    How do you convert timestamp data to date data (datetime datatype)?   
36.    Can I invoke/instantiate COM objects from within stored procedures or triggers using T-SQL?   
37.    Oracle has a rownum to access rows of a table using row number or row id. Is there any equivalent for that in SQL Server? Or How do you generate output with row number in SQL Server?   
38.    How do you specify a network library like TCP/IP using ADO connect string?   
39.    How do you generate scripts for repetitive tasks like truncating all the tables in a database, changing owner of all the database objects, disabling constraints on all tables etc?   
40.    Is there a way to find out when a stored procedure was last updated?   
41.    How do you find out all the IDENTITY columns of all the tables in a given database?   
42.    How do you search the code of stored procedures?   
43.    How do you retrieve the generated GUID value of a newly inserted row? Is there an @@GUID, just like @@IDENTITY?

**1. What are the two authentication modes in SQL Server?**

There are two authentication modes –

* Windows Mode
* Mixed Mode

**2. What Is SQL Profiler?**

SQL Profiler is a tool which allows system administrator to monitor events in the SQL server.

**3. What is recursive stored procedure?**

SQL Server supports recursive stored procedure which calls by itself. Recursive stored procedure can be defined as a method of problem solving wherein the solution is arrived repetitively. It can nest up to 32 levels.

CREATE PROCEDURE [dbo].[Fact]

(

@Number Integer,

@RetVal Integer OUTPUT

)

AS

DECLARE @In Integer

DECLARE @Out Integer

IF @Number != 1

BEGIN

SELECT @In = @Number – 1

EXEC Fact @In, @Out OUTPUT - Same stored procedure has been called again(Recursively)

SELECT @RetVal = @Number \* @Out

END

ELSE

BEGIN

SELECT @RetVal = 1

END

RETURN

GO

**4. What are the differences between local and global temporary tables?**

* Local temporary tables are visible when there is a connection and are deleted when the connection is closed.

CREATE TABLE #<tablename>

* Global temporary tables are visible to all users and are deleted when the connection that created it is closed.

CREATE TABLE ##<tablename>

**5. What is CHECK constraint?**

A CHECK constraint can be applied to a column in a table to limit the values that can be placed in a column. Check constraint is to enforce integrity.

**6. Can SQL servers linked to other servers?**

SQL server can be connected to any database which has OLE-DB provider to give a link. Example: Oracle has OLE-DB provider which has link to connect with the SQL server group.

**7. What is sub query and its properties?**

A sub-query is a query which can be nested inside a main query like Select, Update, Insert or Delete statements. This can be used when expression is allowed. Properties of sub query can be defined as

* A sub query should not have order by clause
* A sub query should be placed in the right-hand side of the comparison operator of the main query
* A sub query should be enclosed in parenthesis because it needs to be executed first before the main query
* More than one sub query can be included

**8. What are the types of sub query?**

There are three types of sub query –

* Single row sub query which returns only one row
* Multiple row sub query which returns multiple rows
* Multiple column sub query which returns multiple columns to the main query. With that sub query result, Main query will be executed.

**9. What is SQL server agent?**

* The SQL Server agent plays a vital role in day to day tasks of SQL server administrator(DBA). Server agent's purpose is to implement the tasks easily with the scheduler engine which allows our jobs to run at scheduled date and time.

**10. What are scheduled tasks in SQL Server?**

Scheduled tasks or jobs are used to automate processes that can be run on a scheduled time at a regular interval. This scheduling of tasks helps to reduce human intervention during night time and feed can be done at a time. User can also order the tasks in which it has to be generated.

**11. What is COALESCE in SQL Server?**

COALESCE is used to return first non-null expression within the arguments.

Select COALESCE(empno, empname, salary) from employee;

**12. How exceptions can be handled in SQL Server Programming?**

Exceptions are handled using TRY----CATCH constructs and it is handles by writing scripts inside the TRY block and error handling in the CATCH block.

**13. What is the purpose of FLOOR function?**

FLOOR function is used to round up a non-integer value to the previous least integer. Example is given

FLOOR(6.7)

Returns 6.

**14. Can we check locks in database? If so, how can we do this lock check?**

Yes, we can check locks in the database. It can be achieved by using in-built stored procedure called sp\_lock.

**15. What is the use of SIGN function?**

SIGN function is used to determine whether the number specified is Positive, Negative and Zero. This will return +1,-1 or 0.

Example –

SIGN(-35) returns -1

**16. What is a Trigger?**

Triggers are used to execute a batch of SQL code when insert or update or delete commands are executed against a table. Triggers are automatically triggered or executed when the data is modified. It can be executed automatically on insert, delete and update operations

**17. What are the types of Triggers?**

There are four types of triggers and they are:

* Insert
* Delete
* Update
* Instead of

**18. What is an IDENTITY column in insert statements?**

IDENTITY column is used in table columns to make that column as Auto incremental number or a surrogate key.

**19. What is Bulkcopy in SQL?**

Bulkcopy is a tool used to copy large amount of data from Tables. This tool is used to load large amount of data in SQL Server.

**20. What will be query used to get the list of triggers in a database?**

Query to get the list of triggers in database-

Select \* from sys.objects where type='tr'

**21. What is the difference between UNION and UNION ALL?**

* UNION: To select related information from two tables UNION command is used.
* UNION All: The UNION ALL command is equal to the UNION command, except that UNION ALL selects all values. It will not remove duplicate rows, instead it will retrieve all rows from all tables

**22. How Global temporary tables are represented and its scope?**

* Global temporary tables are represented with ## before the table name. Scope will be the outside the session whereas local temporary tables are inside the session. Session ID can be found using @@SPID.

**23. What are the differences between Stored Procedure and the dynamic SQL?**

* Stored Procedure is a set of statements which is stored in a compiled form. Dynamic SQL is a set of statements that dynamically constructed at runtime and it will not be stored in a Database and it simply execute during run time.

**24.** **What is Collation?**

Collation is defined to specify the sort order in a table. There are three types of sort order –

1. Case sensitive
2. Case Insensitive
3. Binary

**25. How can we get count of the number of records in a table?**

Following are the queries can be used to get the count of records in a table -

Select \* from <tablename>

Select count(\*) from <tablename>

Select rows from sysindexes

where id=OBJECT\_ID(tablename)

and indid<2

**26. What is the command used to get the version of SQL Server?**

Select SERVERPROPERTY('productversion')

**27. What is UPDATE\_STATISTICS command?**

UPDATE\_STATISTICS command is used to update the indexes on the tables when there is a large amount of deletions or modifications or bulk copy occurred in indexes.

**28. What is the use of SET NOCOUNT ON/OFF statement?**

By default, NOCOUNT is set to OFF and it returns number of records got affected whenever the command is getting executed. If the user doesn't want to display the number of records affected, it can be explicitly set to ON- (SET NOCOUNT ON).

**29. Which SQL server table is used to hold the stored procedure scripts?**

Sys.SQL\_Modules is a SQL Server table used to store the script of stored procedure. Name of the stored procedure is saved in the table called Sys.Procedures.

**30. What are Magic Tables in SQL Server?**

During DML operations like Insert, Delete, and Update, SQL Server creates magic tables to hold the values during the DML operations. These magic tables are used inside the triggers for data transaction.

**31. What is the difference between SUBSTR and CHARINDEX in the SQL Server?**

The SUBSTR function is used to return specific portion of string in a given string. But, CHARINDEX function gives character position in a given specified string.

SUBSTRING('Smiley',1,3)

Gives result as Smi

CHARINDEX('i', 'Smiley',1)

Gives 3 as result as I appears in 3rd position of the string

**32. How can you create a login?**

You can use the following command to create a login

CREATE LOGIN MyLogin WITH PASSWORD = '123';

**33. What is ISNULL() operator?**

ISNULL function is used to check whether value given is NULL or not NULL in sql server. This function also provides to replace a value with the NULL.

**34. What is the use of FOR Clause?**

FOR clause is mainly used for XML and browser options. This clause is mainly used to display the query results in XML format or in browser.

**35. What will be the maximum number of index per table?**

1000 Index can be used as maximum number per table. 1 Clustered Index and 999 Non-clustered indexes per table can be used in SQL Server.

**36. What is the difference between COMMIT and ROLLBACK?**

Every statement between BEGIN and COMMIT becomes persistent to database when the COMMIT is executed. Every statement between BEGIN and ROOLBACK are reverted to the state when the ROLLBACK was executed.

**37. What is the difference between varchar and nvarchar types?**

Varchar and nvarchar are same but the only difference is that nvarhcar can be used to store Unicode characters for multiple languages and it also takes more space when compared with varchar.

**38. What is the use of @@SPID?**

A @@SPID returns the session ID of the current user process.

**39. What is the command used to Recompile the stored procedure at run time?**

Stored Procedure can be executed with the help of keyword called RECOMPILE.

Example

Exe <SPName> WITH RECOMPILE

Or we can include WITHRECOMPILE in the stored procedure itself.

**40. How to delete duplicate rows in SQL Server?**

Duplicate rows can be deleted using CTE and ROW NUMER feature of SQL Server.

**41. Where are SQL Server user names and passwords stored in SQL Server?**

User Names and Passwords are stored in sys.server\_principals and sys.sql\_logins. But passwords are not stored in normal text.

**42. What is the difference between GETDATE and SYSDATETIME?**

Both are same but GETDATE can give time till milliseconds and SYSDATETIME can give precision till nanoseconds. SYSDATE TIME is more accurate than GETDATE.

**43. How data can be copied from one table to another table?**

INSERT INTO SELECT

This command is used to insert data into a table which is already created.

SELECT INTO

This command is used to create a new table and its structure and data can be copied from existing table.

**44. What is TABLESAMPLE?**

TABLESAMPLE is used to extract sample of rows randomly that are all necessary for the application. The sample rows taken are based on the percentage of rows.

**45. Which command is used for user defined error messages?**

RAISEERROR is the command used to generate and initiates error processing for a given session. Those user defined messages are stored in sys.messages table.

**46. What do mean by XML Datatype?**

XML data type is used to store XML documents in the SQL Server database. Columns and variables are created and store XML instances in the database.

**47. What is CDC?**

CDC is abbreviated as Change Data Capture which is used to capture the data that has been changed recently. This feature is present in SQL Server 2008.

**48. What is SQL injection?**

SQL injection is an attack by malicious users in which malicious code can be inserted into strings that can be passed to an instance of SQL server for parsing and execution. All statements have to checked for vulnerabilities as it executes all syntactically valid queries that it receives.

Even parameters can be manipulated by the skilled and experienced attackers.

**49. What are the methods used to protect against SQL injection attack?**

Following are the methods used to protect against SQL injection attack:

* Use Parameters for Stored Procedures
* Filtering input parameters
* Use Parameter collection with Dynamic SQL
* In like clause, user escape characters

**50. What is Filtered Index?**

* Filtered Index is used to filter some portion of rows in a table to improve query performance, index maintenance and reduces index storage costs. When the index is created with WHERE clause, then it is called Filtered Index

**1. What is DBMS?**

* A Database Management System (DBMS) is a program that controls creation, maintenance and use of a database. DBMS can be termed as File Manager that manages data in a database rather than saving it in file systems.
* **2. What is RDBMS?**
* RDBMS stands for Relational Database Management System. RDBMS store the data into the collection of tables, which is related by common fields between the columns of the table. It also provides relational operators to manipulate the data stored into the tables.
* **3. What is SQL?**
* SQL stands for Structured Query Language , and it is used to communicate with the Database. This is a standard language used to perform tasks such as retrieval, updation, insertion and deletion of data from a database.
* Standard SQL Commands are Select.
* **4. What is a Database?**
* Database is nothing but an organized form of data for easy access, storing, retrieval and managing of data. This is also known as structured form of data which can be accessed in many ways.
* Example: School Management Database, Bank Management Database.
* **5. What are tables and Fields?**
* A table is a set of data that are organized in a model with Columns and Rows. Columns can be categorized as vertical, and Rows are horizontal. A table has specified number of column called fields but can have any number of rows which is called record.
* Example:.
* Table: Employee.
* Field: Emp ID, Emp Name, Date of Birth.
* Data: 201456, David, 11/15/1960.
* **6. What is a primary key?**
* A primary key is a combination of fields which uniquely specify a row. This is a special kind of unique key, and it has implicit NOT NULL constraint. It means, Primary key values cannot be NULL.

**7. What is a unique key?**

A Unique key constraint uniquely identified each record in the database. This provides uniqueness for the column or set of columns.

A Primary key constraint has automatic unique constraint defined on it. But not, in the case of Unique Key.

There can be many unique constraint defined per table, but only one Primary key constraint defined per table.

**8. What is a foreign key?**

A foreign key is one table which can be related to the primary key of another table. Relationship needs to be created between two tables by referencing foreign key with the primary key of another table.

**9. What is a join?**

This is a keyword used to query data from more tables based on the relationship between the fields of the tables. Keys play a major role when JOINs are used.

**10. What are the types of join and explain each?**

There are various types of join which can be used to retrieve data and it depends on the relationship between tables.

* **Inner Join.**

Inner join return rows when there is at least one match of rows between the tables.

* **Right Join.**

Right join return rows which are common between the tables and all rows of Right hand side table. Simply, it returns all the rows from the right hand side table even though there are no matches in the left hand side table.

* **Left Join.**

Left join return rows which are common between the tables and all rows of Left hand side table. Simply, it returns all the rows from Left hand side table even though there are no matches in the Right hand side table.

* **Full Join.**

Full join return rows when there are matching rows in any one of the tables. This means, it returns all the rows from the left hand side table and all the rows from the right hand side table.

**11. What is normalization?**

Normalization is the process of minimizing redundancy and dependency by organizing fields and table of a database. The main aim of Normalization is to add, delete or modify field that can be made in a single table.

**12. What is Denormalization.**

DeNormalization is a technique used to access the data from higher to lower normal forms of database. It is also process of introducing redundancy into a table by incorporating data from the related tables.

**13. What are all the different normalizations?**

The normal forms can be divided into 5 forms, and they are explained below -.

* **First Normal Form (1NF):.**

This should remove all the duplicate columns from the table. Creation of tables for the related data and identification of unique columns.

* **Second Normal Form (2NF):.**

Meeting all requirements of the first normal form. Placing the subsets of data in separate tables and Creation of relationships between the tables using primary keys.

* **Third Normal Form (3NF):.**

This should meet all requirements of 2NF. Removing the columns which are not dependent on primary key constraints.

* **Fourth Normal Form (4NF):.**

Meeting all the requirements of third normal form and it should not have multi- valued dependencies.

**14. What is a View?**

A view is a virtual table which consists of a subset of data contained in a table. Views are not virtually present, and it takes less space to store. View can have data of one or more tables combined, and it is depending on the relationship.

**15. What is an Index?**

An index is performance tuning method of allowing faster retrieval of records from the table. An index creates an entry for each value and it will be faster to retrieve data.

insert delete update on views

database level trigger After Create\_table

RAISERROR()

NTILE(10)

SQL-free text Search

**16. What are all the different types of indexes?**

There are three types of indexes -.

* **Unique Index.**

This indexing does not allow the field to have duplicate values if the column is unique indexed. Unique index can be applied automatically when primary key is defined.

* **Clustered Index.**

This type of index reorders the physical order of the table and search based on the key values. Each table can have only one clustered index.

* **NonClustered Index.**

NonClustered Index does not alter the physical order of the table and maintains logical order of data. Each table can have 999 nonclustered indexes.

**17. What is a Cursor?**

A database Cursor is a control which enables traversal over the rows or records in the table. This can be viewed as a pointer to one row in a set of rows. Cursor is very much useful for traversing such as retrieval, addition and removal of database records.

**18. What is a relationship and what are they?**

Database Relationship is defined as the connection between the tables in a database. There are various data basing relationships, and they are as follows:.

* One to One Relationship.
* One to Many Relationship.
* Many to One Relationship.
* Self-Referencing Relationship.

**19. What is a query?**

* A DB query is a code written in order to get the information back from the database. Query can be designed in such a way that it matched with our expectation of the result set. Simply, a question to the Database.

**20. What is subquery?**

* A subquery is a query within another query. The outer query is called as main query, and inner query is called subquery. SubQuery is always executed first, and the result of subquery is passed on to the main query.

**21. What are the types of subquery?**

* There are two types of subquery – Correlated and Non-Correlated.
* A **correlated** subquery cannot be considered as independent query, but it can refer the column in a table listed in the FROM the list of the main query.
* A **Non-Correlated** sub query can be considered as independent query and the output of subquery are substituted in the main query

**22. What is a stored procedure?**

* Stored Procedure is a function consists of many SQL statement to access the database system. Several SQL statements are consolidated into a stored procedure and execute them whenever and wherever required.

**23. What is a trigger?**

* A DB trigger is a code or programs that automatically execute with response to some event on a table or view in a database. Mainly, trigger helps to maintain the integrity of the database.
* Example: When a new student is added to the student database, new records should be created in the related tables like Exam, Score and Attendance tables.

**24. What is the difference between DELETE and TRUNCATE commands?**

* DELETE command is used to remove rows from the table, and WHERE clause can be used for conditional set of parameters. Commit and Rollback can be performed after delete statement.
* TRUNCATE removes all rows from the table. Truncate operation cannot be rolled back.

**25. What are local and global variables and their differences?**

* Local variables are the variables which can be used or exist inside the function. They are not known to the other functions and those variables cannot be referred or used. Variables can be created whenever that function is called.
* Global variables are the variables which can be used or exist throughout the program. Same variable declared in global cannot be used in functions. Global variables cannot be created whenever that function is called.

**26. What is a constraint?**

Constraint can be used to specify the limit on the data type of table. Constraint can be specified while creating or altering the table statement. Sample of constraint are.

* NOT NULL.
* CHECK.
* DEFAULT.
* UNIQUE.
* PRIMARY KEY.
* FOREIGN KEY.
* **27. What is data Integrity?**
* Data Integrity defines the accuracy and consistency of data stored in a database. It can also define integrity constraints to enforce business rules on the data when it is entered into the application or database.
* **28. What is Auto Increment?**
* Auto increment keyword allows the user to create a unique number to be generated when a new record is inserted into the table. AUTO INCREMENT keyword can be used in Oracle and IDENTITY keyword can be used in SQL SERVER.
* Mostly this keyword can be used whenever PRIMARY KEY is used.
* **29. What is the difference between Cluster and Non-Cluster Index?**
* Clustered index is used for easy retrieval of data from the database by altering the way that the records are stored. Database sorts out rows by the column which is set to be clustered index.
* A nonclustered index does not alter the way it was stored but creates a complete separate object within the table. It point back to the original table rows after searching.
* **30. What is Datawarehouse?**
* Datawarehouse is a central repository of data from multiple sources of information. Those data are consolidated, transformed and made available for the mining and online processing. Warehouse data have a subset of data called Data Marts.

**Slowly Changing Dimension.**

* Let me give you a simple, classical example:
* On the 1st of January 2002, Employee 'E1' belongs to Business Unit 'BU1' (that's what would be in your Employee Dimension). This employee has a turnover allocated to him on the Business Unit 'BU1' But on the 2nd of June the Employee 'E1' is muted from Business Unit 'BU1' to Business Unit 'BU2.' The entire new turnovers have to belong to the new Business Unit 'BU2' but the old one should belong to the Business Unit 'BU1.'
* If you used the natural business key 'E1' for your employee within your datawarehouse everything would be allocated to Business Unit 'BU2' even what actually belongs to 'BU1.'
* If you use surrogate keys, you could create on the 2nd of June a new record for the Employee 'E1' in your Employee Dimension with a new surrogate key.
* This way, in your fact table, you have your old data (before 2nd of June) with the SID of the Employee 'E1' + 'BU1.' All new data (after 2nd of June) would take the SID of the employee 'E1' + 'BU2.'
* You could consider Slowly Changing Dimension as an enlargement of your natural key: natural key of the Employee was Employee Code 'E1' but for you it becomes
* Employee Code + Business Unit - 'E1' + 'BU1' or 'E1' + 'BU2.' But the difference with the natural key enlargement process, is that you might not have all part of your new key within your fact table, so you might not be able to do the join on the new enlarge key -> so you need another id.

**Fact table and dimension table difference**

* The fact table mainly consists of business facts and foreign keys that refer to primary keys in the dimension tables. A dimension table consists mainly of descriptive attributes that are textual fields.
* A dimension table contains a surrogate key, natural key, and a set of attributes. On the contrary, a fact table contains a foreign key, measurements, and degenerated dimensions.
* A fact table normally contains two columns – one that contains the facts and the other foreign keys to the dimension table. Dimension tables are also called reference tables.
* **31. What is Self-Join?**
* Self-join is set to be query used to compare to itself. This is used to compare values in a column with other values in the same column in the same table. ALIAS ES can be used for the same table comparison.
* **32. What is Cross-Join?**
* Cross join defines as Cartesian product where number of rows in the first table multiplied by number of rows in the second table. If suppose, WHERE clause is used in cross join then the query will work like an INNER JOIN.
* **33. What is user defined functions?**
* User defined functions are the functions written to use that logic whenever required. It is not necessary to write the same logic several times. Instead, function can be called or executed whenever needed.

**34. What are all types of user defined functions?**

Three types of user defined functions are.

* Scalar Functions.
* Inline Table valued functions.
* Multi statement valued functions.

Scalar returns unit, variant defined the return clause. Other two types return table as a return.

**35. What is collation?**

Collation is defined as set of rules that determine how character data can be sorted and compared. This can be used to compare A and, other language characters and also depends on the width of the characters.

ASCII value can be used to compare these character data.

**36. What are all different types of collation sensitivity?**

Following are different types of collation sensitivity -.

* Case Sensitivity – A and a and B and b.
* Accent Sensitivity.
* Kana Sensitivity – Japanese Kana characters.
* Width Sensitivity – Single byte character and double byte character.

**37. Advantages and Disadvantages of Stored Procedure?**

* Stored procedure can be used as a modular programming – means create once, store and call for several times whenever required. This supports faster execution instead of executing multiple queries. This reduces network traffic and provides better security to the data.
* Disadvantage is that it can be executed only in the Database and utilizes more memory in the database server.

**38. What is Online Transaction Processing (OLTP)?**

* Online Transaction Processing (OLTP) manages transaction based applications which can be used for data entry, data retrieval and data processing. OLTP makes data management simple and efficient. Unlike OLAP systems goal of OLTP systems is serving real-time transactions.
* Example – Bank Transactions on a daily basis.

**39. What is CLAUSE?**

* SQL clause is defined to limit the result set by providing condition to the query. This usually filters some rows from the whole set of records.
* Example – Query that has WHERE condition
* Query that has HAVING condition.

**40. What is recursive stored procedure?**

* A stored procedure which calls by itself until it reaches some boundary condition. This recursive function or procedure helps programmers to use the same set of code any number of times.
* **41. What is Union, minus and Interact commands?**
* UNION operator is used to combine the results of two tables, and it eliminates duplicate rows from the tables.
* MINUS operator is used to return rows from the first query but not from the second query. Matching records of first and second query and other rows from the first query will be displayed as a result set.
* INTERSECT operator is used to return rows returned by both the queries.

**42. What is an ALIAS command?**

* ALIAS name can be given to a table or column. This alias name can be referred in WHERE clause to identify the table or column.
* Example-.
* Select st.StudentID, Ex.Result from student st, Exam as Ex where st.studentID = Ex. StudentID

**43. What is the difference between TRUNCATE and DROP statements?**

* TRUNCATE removes all the rows from the table, and it cannot be rolled back. DROP command removes a table from the database and operation cannot be rolled back.

**44. What are aggregate and scalar functions?**

* Aggregate functions are used to evaluate mathematical calculation and return single values. This can be calculated from the columns in a table. Scalar functions return a single value based on the input value.
* Example -.
* Aggregate – max(), count - Calculated with respect to numeric.
* Scalar – UCASE(), NOW() – Calculated with respect to strings.
* **45. How can you create an empty table from an existing table?**
* Example will be -.
* Select \* into studentcopy
* from student where 1=2

**46. How to fetch common records from two tables?**

* Common records result set can be achieved by -.
* Select studentID from student.
* INTERSECT
* Select StudentID from Exam

**47. How to fetch alternate records from a table?**

* Records can be fetched for both Odd and Even row numbers -.
* To display even numbers-.
* Select studentId from (Select rowno, studentId from student) where mod(rowno,2)=0
* To display odd numbers-.
* Select studentId from (Select rowno, studentId from student) where mod(rowno,2)=1

**48. How to select unique records from a table?**

* Select unique records from a table by using DISTINCT keyword.
* Select DISTINCT StudentID, StudentName from Student.

**49. What is the command used to fetch first 5 characters of the string?**

* There are many ways to fetch first 5 characters of the string -.
* Select SUBSTRING(StudentName,1,5) as studentname from student
* Select LEFT(Studentname,5) as studentname from student

**50. Which operator is used in query for pattern matching?**

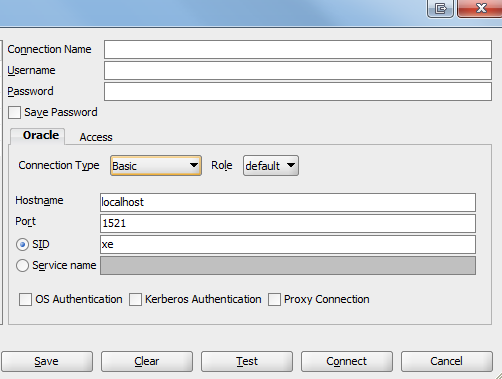
LIKE operator is used for pattern matching, and it can be used as -.

1. % - Matches zero or more characters.
2. \_(Underscore) – Matching exactly one character.

**Example -.**

Select \* from Student where studentname like 'a%'

Select \* from Student where studentname like 'ami\_'



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

-- :::: Primary Key

-- SQL::::

SELECT distinct upper(A.Name)+'\_'+upper(Col.Column\_Name) as keycols,

upper(A.Name) as Table\_Name,upper(Col.Column\_Name) as Column\_Name, Col.Constraint\_Name from

INFORMATION\_SCHEMA.TABLE\_CONSTRAINTS Tab,

INFORMATION\_SCHEMA.CONSTRAINT\_COLUMN\_USAGE Col ,

(select NAME from dbo.sysobjects where xtype='u') AS A

WHERE

Col.Constraint\_Name = Tab.Constraint\_Name

AND Col.Table\_Name = Tab.Table\_Name

AND Constraint\_Type = 'PRIMARY KEY '

AND Col.Table\_Name = A.Name

order by 1,2

-- Oracle::::

select distinct all\_cols.table\_name ||'\_'|| all\_cols.column\_name as keycol

,ALL\_COLS.TABLE\_NAME

,ALL\_COLS.COLUMN\_NAME

, all\_cols.constraint\_name as Primary\_Key

from all\_cons\_columns col

inner join all\_cons\_columns all\_cols

on col.owner = all\_cols.owner

and col.constraint\_name = all\_cols.constraint\_name

inner join all\_constraints cons

on col.owner = cons.owner

and COL.CONSTRAINT\_NAME = CONS.CONSTRAINT\_NAME

where col.owner = '<Owner\_Name>'

and CONS.CONSTRAINT\_TYPE in ('P')

order by 2,4;

-- :::: Foreign Key

-- SQL::::

SELECT

o1.name+'\_'+c1.name as keycol,

o1.name AS Table\_Name,

fk.name AS FK\_Name,

c1.name AS FK\_Column,

case when is\_disabled = 0 then 'ENABLED' when is\_disabled = 1 then 'DISABLED' END as STATUS

FROM sys.objects o1

INNER JOIN sys.foreign\_keys fk

ON o1.object\_id = fk.parent\_object\_id

INNER JOIN sys.foreign\_key\_columns fkc

ON fk.object\_id = fkc.constraint\_object\_id

INNER JOIN sys.columns c1

ON fkc.parent\_object\_id = c1.object\_id

AND fkc.parent\_column\_id = c1.column\_id

INNER JOIN sys.columns c2

ON fkc.referenced\_object\_id = c2.object\_id

AND fkc.referenced\_column\_id = c2.column\_id

INNER JOIN sys.objects o2

ON fk.referenced\_object\_id = o2.object\_id

INNER JOIN sys.key\_constraints pk

ON fk.referenced\_object\_id = pk.parent\_object\_id

AND fk.key\_index\_id = pk.unique\_index\_id

ORDER BY o1.name, c1.name

-- Oracle::::

select all\_cols.table\_name ||'\_'|| all\_cols.column\_name as keycol

,ALL\_COLS.TABLE\_NAME

, ALL\_COLS.CONSTRAINT\_NAME AS FK\_NAME

,ALL\_COLS.COLUMN\_NAME as FK\_Column

,CONS.status

from all\_cons\_columns col

inner join all\_cons\_columns all\_cols

on col.owner = all\_cols.owner

and col.constraint\_name = all\_cols.constraint\_name

inner join all\_constraints cons

on col.owner = cons.owner

and col.constraint\_name = cons.constraint\_name

where col.owner = '<Owner\_Name>'

AND CONS.CONSTRAINT\_TYPE IN ('R')

order by all\_cols.Table\_name;

-- :::: Default CONSTRAINTS

-- SQL ::::

SELECT

t.name +'\_'+ c.name as KeyCol,

t.name as Table\_Name,

c.name as column\_name,

dc.name as DefaultConstraint\_Name

FROM

sys.all\_columns c

INNER JOIN

sys.tables t

ON c.object\_id = t.object\_id

INNER JOIN

sys.schemas s

ON t.schema\_id = s.schema\_id

INNER JOIN

sys.default\_constraints dc

ON c.default\_object\_id = dc.object\_id

WHERE

s.name = 'dbo'

order by t.name,c.name

-- Oracle ::::

select table\_name || '\_' || column\_name,table\_name,column\_name,data\_default

FROM ALL\_TAB\_COLUMNS

where OWNER = 'PROLOGIS\_PM'

AND DATA\_DEFAULT IS NOT NULL

order by table\_name

-- :::: UNIQUE CONSTRAINTS

-- SQL ::::

select TC.TABLE\_NAME+'\_'+CC.Column\_Name as KeyCols,

TC.TABLE\_NAME, TC.Constraint\_Name, CC.Column\_Name

from information\_schema.table\_constraints TC

inner join information\_schema.constraint\_column\_usage CC

on TC.Constraint\_Name = CC.Constraint\_Name

where TC.constraint\_type = 'Unique'

and TC.CONSTRAINT\_NAME not like 'PK\_%'

order by 2,4

-- Oracle ::::

select distinct all\_cols.table\_name ||'\_'|| all\_cols.column\_name as keycol

,ALL\_COLS.TABLE\_NAME

,ALL\_COLS.COLUMN\_NAME

, all\_cols.constraint\_name as Unique\_Key

from all\_cons\_columns col

inner join all\_cons\_columns all\_cols

on col.owner = all\_cols.owner

and col.constraint\_name = all\_cols.constraint\_name

inner join all\_constraints cons

on col.owner = cons.owner

and COL.CONSTRAINT\_NAME = CONS.CONSTRAINT\_NAME

where col.owner = '<Owner\_Name>'

and CONS.CONSTRAINT\_TYPE in ('U')

order by 2,4;

-- Index

--SQL -

with abc as (

SELECT

Ind.[name] AS IndexName,

Tab.name TableName,

SUBSTRING(( SELECT ',' + AC.name

FROM sys.[tables] AS T

INNER JOIN sys.[indexes] I

ON T.[object\_id] = I.[object\_id]

INNER JOIN sys.[index\_columns] IC

ON I.[object\_id] = IC.[object\_id]

AND I.[index\_id] = IC.[index\_id]

INNER JOIN sys.[all\_columns] AC

ON T.[object\_id] = AC.[object\_id]

AND IC.[column\_id] = AC.[column\_id]

WHERE Ind.[object\_id] = I.[object\_id]

AND Ind.index\_id = I.index\_id

AND IC.is\_included\_column = 0

ORDER BY AC.name

FOR XML PATH('')

), 2, 8000) AS KeyCols

FROM sys.[indexes] Ind

INNER JOIN sys.[tables] AS Tab

ON Tab.[object\_id] = Ind.[object\_id]

INNER JOIN sys.[schemas] AS Sch

ON Sch.[schema\_id] = Tab.[schema\_id]

where

ind.is\_primary\_key = 0 -- do not include PK indexes

and ind.is\_unique\_constraint = 0 -- do not include UQ

and Ind.[name] is not null

)

select tablename+'\_'+KeyCols as keys, tablename, IndexName,KeyCols from abc

ORDER BY 1,2

--ORACLE -

with abc as (

select a.INDEX\_NAME ,a.TABLE\_NAME,

listagg(b.column\_name, ',' ) WITHIN GROUP (order by b.column\_name,a.table\_name,a.index\_name) over(partition by A.TABLE\_NAME,a.index\_name)

Keycolumn

from SYS.ALL\_INDEXES A

JOIN all\_ind\_columns B

ON A.INDEX\_NAME = B.INDEX\_NAME

WHERE A.TABLE\_OWNER = 'PROLOGIS\_ABF'

GROUP BY A.TABLE\_NAME,A.INDEX\_NAME,B.COLUMN\_NAME)

SELECT DISTINCT TABLE\_NAME||'\_'||KEYCOLUMN AS KEYSC, TABLE\_NAME, INDEX\_NAME, KEYCOLUMN FROM ABC

where TABLE\_NAME||'\_'||KEYCOLUMN not in (

select COL.TABLE\_NAME||'\_'||

LISTAGG(COL.COLUMN\_NAME, ',' )

within group (order by COL.TABLE\_NAME,COL.COLUMN\_NAME)

over(partition by COL.TABLE\_NAME)

as cols

from all\_cons\_columns col

inner join all\_constraints cons

on col.owner = cons.owner

and COL.CONSTRAINT\_NAME = CONS.CONSTRAINT\_NAME

and COL.TABLE\_NAME = CONS.TABLE\_NAME

where COL.OWNER = 'PROLOGIS\_ABF'

and CONS.CONSTRAINT\_TYPE in ('P')

union

select COL.TABLE\_NAME||'\_'||

LISTAGG(COL.COLUMN\_NAME, ',' )

within group (order by COL.TABLE\_NAME,COL.COLUMN\_NAME)

over(partition by COL.TABLE\_NAME, CONS.CONSTRAINT\_NAME)

as cols

from all\_cons\_columns col

inner join all\_constraints cons

on col.owner = cons.owner

and COL.CONSTRAINT\_NAME = CONS.CONSTRAINT\_NAME

and COL.TABLE\_NAME = CONS.TABLE\_NAME

where COL.OWNER = 'PROLOGIS\_ABF'

and CONS.CONSTRAINT\_TYPE in ('U')

)

--and TABLE\_NAME = 'DCFJOURNAL'

order by 2,4;

----- Stored Procedure::::

---- Oracle::

select object\_name from ALL\_OBJECTS

where OWNER = '<Owner>' and OBJECT\_TYPE = 'PROCEDURE'

order by object\_name ;

----SQL::::

select upper(name) as Stored\_Procedure\_Name FROM sys.sql\_modules m

INNER JOIN sys.objects o ON m.object\_id=o.object\_id

where type = 'P'

order by name;

---------Trigger:::::

------ Oracle::

select object\_name from ALL\_OBJECTS

where owner = '<Owner>' and object\_type = 'TRIGGER'

order by object\_name ;

----SQL::::

select upper(name) as Trigger\_Name FROM sys.sql\_modules m

INNER JOIN sys.objects o ON m.object\_id=o.object\_id

where type = 'TR'

order by name;

--ORACLE FUNCTIONS

select object\_name from all\_objects

where owner = <OWNER>

and object\_type = 'FUNCTION'

order by object\_name

--SQL FUNCTIONS

select NAME FROM sys.sql\_modules m

INNER JOIN sys.objects o ON m.object\_id=o.object\_id

where TYPE = 'FN'

order by name

--ORACLE VIEWS

select object\_name from all\_objects

where owner = ‘<Owner>'

and object\_type = 'VIEW'

order by object\_name;

--SQL VIEWS

select upper(name) Name FROM sys.sql\_modules m

INNER JOIN sys.objects o ON m.object\_id=o.object\_id

where TYPE = 'V'

order by name

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

nvl ----------> isnull

trim( ----------> ltrim(rtrim(

to\_Char(isnull(t.dtMoveOut, t.dtLeaseTo), 'YYYY')

-----> select DATEPART(yyyy,getdate())

add\_months(#yearend#, - 11)

last\_day(#yearend#) dbo.

CONVERT(VARCHAR,(tr.hmy - 700000000)) ------> To\_Char(tr.hmy - 700000000)

'C-' || CONVERT --------> 'C-' + CONVERT

TO\_NUMBER(REPLACE(AREA.LANDHECTARE, N',', ''))

-----------> Convert(Numeric, REPLACE(AREA.LANDHECTARE, N',', ''))

to\_date(leaseto, 'dd/mm/yy') ----> Select convert(varchar(10),leaseto,103)

(select nvl(cd.Addr1,'') || nvl(cd.Addr2,'') Address from contact\_details cd where cd.itype=2 and cd.contactrole ='Signature Block' and cd.hperson = OwnerId and rownum = 1)

---------->

( SELECT top 1 isnull(cd.Addr1, '') + isnull(cd.Addr2, '') Address

FROM contact\_details cd

WHERE cd.itype = 2

AND cd.contactrole = N'Signature Block'

AND cd.hperson = OwnerId

order by 1

)

date1 - date2 ----------------------> datediff(day, small, large)

length ------------> len

TRUNC (cr.dtincrease) <= getdate() -----------> convert(date,getdate())

SELECT ROWNUM rn FROM ( SELECT 1,2,3,4,5 FROM DUAL GROUP BY CUBE(1, 2, 3, 4, 5) ) WHERE ROWNUM <= 19

----------->

SELECT TOP 19 row\_number() OVER (ORDER BY NAME) onum FROM sysobjects

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

convert(date,getdate()) + 365 --->

Select DateAdd(Day, 365, convert(Date,getdate()))

CAST (NVL (sqft.sf, 0) AS NUMBER (21, 6)) LEASE\_AREA\_SF

------->

cast( isnull(sqft.sf,0) as float) LEASE\_AREA\_SF

------------------------->

Set IDENTITY\_INSERT T\_Tasks ON

INSERT INTO T\_TASKS (hmy, hTSTask, hYardiSchedule, sName, sDesc, bInactive, dtLastRun, iStatus, iTimeOut, iPriority, bNotifyOnFailure, sNotifyOnFailureList, bNotifyOnSuccess, sNotifyOnSuccessList, bRunNow )

SELECT hmy, hTSTask, hYardiSchedule, sName, sDesc, bInactive, dtLastRun, iStatus, iTimeOut, iPriority, bNotifyOnFailure, sNotifyOnFailureList, bNotifyOnSuccess, sNotifyOnSuccessList, bRunNow

FROM tsschedule WHERE sname LIKE N'%sys00%'

AND istatus = 1

AND BINACTIVE = - 1;

------------------->PLD\_GET\_EXCL\_ACCT

select \* from all\_source where upper(name) = upper('PLD\_UPDATERECOVERY') and upper(owner) = upper('yrykz\_yadpcm\_101915')

select max(

max(ltrim(sys\_connect\_by\_path(rtrim(scode),','),',')) keep

(dense\_rank last order by level) )as excl\_acct

into sacct

from (

select distinct cr2.hmy crhmy,a.scode,row\_number() over(partition by cr2.hmy order by cr2.hmy,a.scode) as rn

from commrecoveryexclude cx inner join camrule cr2 on cx.HCAMRULE=cr2.hmy inner join acct a on cx.hacct=a.hmy

Where cr2.hmy= crhmy

)

start with rn = 1

connect by prior crhmy = crhmy

and prior rn = rn - 1

group by crhmy,scode;

--------------------------->

SELECT @sacct = REVERSE(STUFF(REVERSE((

select distinct ltrim(rtrim(a.scode)) + ','

from commrecoveryexclude cx inner join camrule cr2 on cx.HCAMRULE=cr2.hmy inner join acct a on cx.hacct=a.hmy

Where cr2.hmy= @crhmy

ORDER BY ltrim(rtrim(a.scode)) + ','

FOR XML PATH(''))), 1, 1, ''))

To\_Char('#asofdate#', 'dd-MON-YYYY') --> CONVERT(varchar,'#asofdate#', 106);

execute immediate -----> sp\_executesql

SELECT LEAST

(

select expression1 from dual,

select expression2 from dual,

select expression3 from dual

) from dual

-------------->

SELECT min(c1)

from (

select expression1 as c1

union all

select expression2 as c1

union all

select expression3 as c1

)

IF THEN END IF; -------> IF BEGIN END;

DECLARE

c INT;

BEGIN

SELECT COUNT (\*)

INTO c

FROM user\_tables

WHERE table\_name = UPPER ('Temp\_HistoricalRent');

IF c = 1

THEN

EXECUTE IMMEDIATE 'DROP TABLE Temp\_HistoricalRent';

END IF;

END;

----------->

IF EXISTS (SELECT 1 FROM sysObjects WHERE 1=1 AND NAME = 'Temp\_HistoricalRent' AND XTYPE = 'U')

DROP TABLE Temp\_HistoricalRent

END