Table Name : Employee

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EMPLOYEE\_ID | FIRST\_NAME | LAST\_NAME | SALARY | JOINING\_DATE | DEPARTMENT |
| 1 | John | Abraham | 1000000 | 01-JAN-13 12.00.00 AM | Banking |
| 2 | Michael | Clarke | 800000 | 01-JAN-13 12.00.00 AM | Insurance |
| 3 | Roy | Thomas | 700000 | 01-FEB-13 12.00.00 AM | Banking |
| 4 | Tom | Jose | 600000 | 01-FEB-13 12.00.00 AM | Insurance |
| 5 | Jerry | Pinto | 650000 | 01-FEB-13 12.00.00 AM | Insurance |
| 6 | Philip | Mathew | 750000 | 01-JAN-13 12.00.00 AM | Services |
| 7 | TestName1 | 123 | 650000 | 01-JAN-13 12.00.00 AM | Services |
| 8 | TestName2 | Lname% | 600000 | 01-FEB-13 12.00.00 AM | Insurance |

Table Name : Incentives

|  |  |  |
| --- | --- | --- |
| EMPLOYEE\_REF\_ID | INCENTIVE\_DATE | INCENTIVE\_AMOUNT |
| 1 | 01-FEB-13 | 5000 |
| 2 | 01-FEB-13 | 3000 |
| 3 | 01-FEB-13 | 4000 |
| 1 | 01-JAN-13 | 4500 |
| 2 | 01-JAN-13 | 3500 |

# SQL Select - Examples

**1. Get all employee details from the employee table**

Select \* from employee

**2. Get First\_Name,Last\_Name from employee table**

Select first\_name, Last\_Name from employee

**3. Get First\_Name from employee table using alias name “Employee Name”**

Select first\_name Employee Name from employee

**4. Get First\_Name from employee table in upper case**

Select upper(FIRST\_NAME) from EMPLOYEE

**5. Get First\_Name from employee table in lower case**

Select lower(FIRST\_NAME) from EMPLOYEE

**6. Get unique DEPARTMENT from employee table**

select distinct DEPARTMENT from EMPLOYEE

**7. Select first 3 characters of FIRST\_NAME from EMPLOYEE**

Oracle Equivalent of SQL Server SUBSTRING is SUBSTR, Query :

select substr(FIRST\_NAME,0,3) from employee

SqL Server Equivalent of Oracle SUBSTR is SUBSTRING, Query :

select substring(FIRST\_NAME,0,3) from employee

MySQL Server Equivalent of Oracle SUBSTR is SUBSTRING. In MySQL start position is 1, Query :

select substring(FIRST\_NAME,1,3) from employee

**8. Get position of 'o' in name 'John' from employee table**

Oracle Equivalent of SQL Server CHARINDEX is INSTR,

Query : Select instr(FIRST\_NAME,'o') from employee where first\_name='John'

SQL Server Equivalent of Oracle INSTR is CHARINDEX,

Query: Select CHARINDEX('o',FIRST\_NAME,0) from employee where first\_name='John'

MySQL Server Equivalent of Oracle INSTR is LOCATE,

Query: Select LOCATE('o',FIRST\_NAME) from employee where first\_name='John'

**9. Get FIRST\_NAME from employee table after removing white spaces from right side**

select RTRIM(FIRST\_NAME) from employee

**10. Get FIRST\_NAME from employee table after removing white spaces from left side**

select LTRIM(FIRST\_NAME) from employee

**11. Get length of FIRST\_NAME from employee table**

Oracle,MYSQL Equivalent of SQL Server Len is Length ,

Query :select length(FIRST\_NAME) from employee

SQL Server Equivalent of Oracle,MYSQL Length is Len,

Query :select len(FIRST\_NAME) from employee

**12. Get First\_Name from employee table after replacing 'o' with '$'**

select REPLACE(FIRST\_NAME,'o','$') from employee

**13. Get First\_Name and Last\_Name as single column from employee table separated by a '\_'**

Oracle Equivalent of MySQL concat is '||',

Query : Select FIRST\_NAME|| '\_' ||LAST\_NAME from EMPLOYEE  
**SQL Server Equivalent of MySQL concat is '+'**,

Query : Select FIRST\_NAME + '\_' +LAST\_NAME from EMPLOYEE  
**MySQL Equivalent of Oracle '||' is concat**,

Query : Select concat(FIRST\_NAME,'\_',LAST\_NAME) from EMPLOYEE

**14. Get FIRST\_NAME ,Joining year,Joining Month and Joining Date from employee table**

SQL Queries in Oracle,

Select FIRST\_NAME, to\_char(joining\_date,'YYYY') JoinYear , to\_char(joining\_date,'Mon'), to\_char(joining\_date,'dd') from EMPLOYEE  
**SQL Queries in SQL Server**,

select SUBSTRING (convert(varchar,joining\_date,103),7,4) , SUBSTRING (convert(varchar,joining\_date,100),1,3) , SUBSTRING (convert(varchar,joining\_date,100),5,2) from EMPLOYEE  
**SQL Queries in MySQL**,

select year(joining\_date),month(joining\_date), DAY(joining\_date) from EMPLOYEE

# ****SQL Order By****

**15. Get all employee details from the employee table order by First\_Name Ascending**

**Select \* from employee order by FIRST\_NAME asc**

**16. Get all employee details from the employee table order by First\_Name descending**

**Select \* from employee order by FIRST\_NAME desc**

**17. Get all employee details from the employee table order by First\_Name Ascending and Salary descending**

Select \* from employee order by FIRST\_NAME asc,SALARY desc

# ****SQL Where Condition - Examples****

**18. Get employee details from employee table whose employee name is “John”**

Select \* from EMPLOYEE where FIRST\_NAME='John'

**19. Get employee details from employee table whose employee name are “John” and “Roy”**

Select \* from EMPLOYEE where FIRST\_NAME in ('John','Roy')

**20. Get employee details from employee table whose employee name are not “John” and “Roy”**

Select \* from EMPLOYEE where FIRST\_NAME not in ('John','Roy')

# ****SQL Wild Card Search - Examples****

**21. Get employee details from employee table whose first name starts with 'J'**

Select \* from EMPLOYEE where FIRST\_NAME like 'J%'

**22. Get employee details from employee table whose first name contains 'o'**

Select \* from EMPLOYEE where FIRST\_NAME like '%o%'

**23. Get employee details from employee table whose first name ends with 'n'**

Select \* from EMPLOYEE where FIRST\_NAME like '%n'

# ****SQL Pattern Matching - Examples****

**24. Get employee details from employee table whose first name ends with 'n' and name contains 4 letters**

Select \* from EMPLOYEE where FIRST\_NAME like '\_\_\_n' (Underscores)

**25. Get employee details from employee table whose first name starts with 'J' and name contains 4 letters**

Select \* from EMPLOYEE where FIRST\_NAME like 'J\_\_\_' (Underscores)

**26. Get employee details from employee table whose Salary greater than 600000**

Select \* from EMPLOYEE where Salary >600000

**27. Get employee details from employee table whose Salary less than 800000**

Select \* from EMPLOYEE where Salary <800000

**28. Get employee details from employee table whose Salary between 500000 and 800000**

Select \* from EMPLOYEE where Salary between 500000 and 800000

**29. Get employee details from employee table whose name is 'John' and 'Michael'**

Select \* from EMPLOYEE where FIRST\_NAME in ('John','Michael')

Functions" - Examples

**30. Get employee details from employee table whose joining year is “2013”**

**SQL Queries in Oracle**, Select \* from EMPLOYEE where to\_char(joining\_date,'YYYY')='2013'  
  
**SQL Queries in SQL Server**, Select \* from EMPLOYEE where SUBSTRING(convert(varchar,joining\_date,103),7,4)='2013'  
  
**SQL Queries in MySQL**, Select \* from EMPLOYEE where year(joining\_date)='2013'

**31. Get employee details from employee table whose joining month is “January”**

SQL Queries in Oracle, Select \* from EMPLOYEE where to\_char(joining\_date,'MM')='01' or Select \* from EMPLOYEE where to\_char(joining\_date,'Mon')='Jan'

SQL Queries in SQL Server, Select \* from EMPLOYEE where SUBSTRING(convert(varchar,joining\_date,100),1,3)='Jan'

SQL Queries in MySQL, Select \* from EMPLOYEE where month(joining\_date)='01'

**32. Get employee details from employee table who joined before January 1st 2013**

SQL Queries in Oracle, Select \* from EMPLOYEE where JOINING\_DATE <to\_date('01/01/2013','dd/mm/yyyy')  
SQL Queries in SQL Server (Format - “MM/DD/YYYY”), Select \* from EMPLOYEE where joining\_date <'01/01/2013'  
SQL Queries in MySQL (Format - “YYYY-DD-MM”), Select \* from EMPLOYEE where joining\_date <'2013-01-01'

**33. Get employee details from employee table who joined after January 31st**

SQL Queries in Oracle, Select \* from EMPLOYEE where JOINING\_DATE >to\_date('31/01/2013','dd/mm/yyyy')  
SQL Queries in SQL Server and MySQL (Format - “MM/DD/YYYY”), Select \* from EMPLOYEE where joining\_date >'01/31/2013'  
SQL Queries in MySQL (Format - “YYYY-DD-MM”), Select \* from EMPLOYEE where joining\_date >'2013-01-31'

**35. Get Joining Date and Time from employee table**

SQL Queries in Oracle,

select to\_char(JOINING\_DATE,'dd/mm/yyyy hh:mi:ss') from EMPLOYEE  
**SQL Queries in SQL Server,**

Select convert(varchar(19),joining\_date,121) from EMPLOYEE  
**SQL Queries in MySQL,**

Select CONVERT(DATE\_FORMAT(joining\_date,'%Y-%m-%d-%H:%i:00'),DATETIME) from EMPLOYEE

**36. Get Joining Date,Time including milliseconds from employee table**

SQL Queries in Oracle,

select to\_char(JOINING\_DATE,'dd/mm/yyyy HH:mi:ss.ff') from EMPLOYEE . Column Data Type should be “TimeStamp”  
**SQL Queries in SQL Server,**

select convert(varchar,joining\_date,121) from EMPLOYEE  
**SQL Queries in MySQL,**

Select MICROSECOND(joining\_date) from EMPLOYEE

**37. Get difference between JOINING\_DATE and INCENTIVE\_DATE from employee and incentives table**

Select FIRST\_NAME,INCENTIVE\_DATE - JOINING\_DATE from employee a inner join incentives B on A.EMPLOYEE\_ID=B.EMPLOYEE\_REF\_ID

**38. Get database date**

SQL Queries in Oracle, select sysdate from dual  
SQL Queries in SQL Server, select getdate()  
SQL Query in MySQL, select now()

# ****Escape Characters - Examples****

**39. Get names of employees from employee table who has '%' in Last\_Name. Tip : Escape character for special characters in a query.**

**SQLQueries in Oracle, Select FIRST\_NAME from employee where Last\_Name like '%?%%'**

**SQLQueries in SQL Server, Select FIRST\_NAME from employee where Last\_Name like '%[%]%'**

**SQLQueries in MySQL,Select FIRST\_NAME from employee where Last\_Name like '%\%%'**

**40. Get Last Name from employee table after replacing special character with white space**

SQL Queries in Oracle,

Select translate(LAST\_NAME,'%',' ') from employee

SQL Queries in SQL Server and MySQL, Select REPLACE(LAST\_NAME,'%',' ') from employee

# ****SQL Group By Functions – Examples****

**41. Get department,total salary with respect to a department from employee table.**

Select DEPARTMENT,sum(SALARY) Total\_Salary from employee group by department

**42. Get department,total salary with respect to a department from employee table order by total salary descending**

Select DEPARTMENT,sum(SALARY) Total\_Salary from employee group by DEPARTMENT order by Total\_Salary descending

# ****SQL Mathematical Operations using Group By - Examples****

**43. Get department,no of employees in a department,total salary with respect to a department from employee table order by total salarydescending**

Select DEPARTMENT,count(FIRST\_NAME),sum(SALARY) Total\_Salary from employee group by DEPARTMENT order by Total\_Salary descending

**44. Get department wise average salary from employee table order by salaryascending**

select DEPARTMENT,avg(SALARY) AvgSalary from employee group by DEPARTMENT order by AvgSalary asc

**45. Get department wise maximum salary from employee table order by salaryascending**

select DEPARTMENT,max(SALARY) MaxSalary from employee group by DEPARTMENT order by MaxSalary asc

**46. Get department wise minimum salary from employee table order by salary ascending**

select DEPARTMENT,min(SALARY) MinSalary from employee group by DEPARTMENT order by MinSalary asc

**47. Select no of employees joined with respect to year and month from employee table**

**SQL Queries in Oracle,** select to\_char (JOINING\_DATE,'YYYY') Join\_Year,to\_char (JOINING\_DATE,'MM') Join\_Month,count(\*) Total\_Emp from employee group by to\_char (JOINING\_DATE,'YYYY'),to\_char(JOINING\_DATE,'MM')

**SQL Queries in SQL Server,** select datepart (YYYY,JOINING\_DATE) Join\_Year,datepart (MM,JOINING\_DATE) Join\_Month,count(\*) Total\_Emp from employee group by datepart(YYYY,JOINING\_DATE), datepart(MM,JOINING\_DATE)

**SQL Queries in MySQL,** select year (JOINING\_DATE) Join\_Year,month (JOINING\_DATE) Join\_Month,count(\*) Total\_Emp from employee group by year(JOINING\_DATE), month(JOINING\_DATE)

**48. Select department,total salary with respect to a department from employee table where total salary greater than 800000 order by Total\_Salary descending**

Select DEPARTMENT,sum(SALARY) Total\_Salary from employee group by DEPARTMENT having sum(SALARY) >800000 order by Total\_Salary desc

# ****Join- Examples****

**49. Select first\_name, incentive amount from employee and incentives table for those employees who have incentives**

**Select FIRST\_NAME,INCENTIVE\_AMOUNT from employee A inner join incentives B on A.EMPLOYEE\_ID=B.EMPLOYEE\_REF\_ID**

**50. Select first\_name, incentive amount from employee and incentives table for those employees who have incentives and incentive amount greater than 3000**

Select FIRST\_NAME,INCENTIVE\_AMOUNT from employee a inner join incentives B on A.EMPLOYEE\_ID=B.EMPLOYEE\_REF\_ID and INCENTIVE\_AMOUNT >3000

**51. Select first\_name, incentive amount from employee and incentives table for all employes even if they didn't get incentives**

Select FIRST\_NAME,INCENTIVE\_AMOUNT from employee a left join incentives B on A.EMPLOYEE\_ID=B.EMPLOYEE\_REF\_ID

**52. Select first\_name, incentive amount from employee and incentives table for all employees even if they didn't get incentives and set incentive amount as 0 for those employees who didn't get incentives.**

**SQL Queries in Oracle,** Select FIRST\_NAME,nvl(INCENTIVE\_AMOUNT,0) from employee a left join incentives B on A.EMPLOYEE\_ID=B.EMPLOYEE\_REF\_ID  
**SQL Queries in SQL Server,** Select FIRST\_NAME, ISNULL(INCENTIVE\_AMOUNT,0) from employee a left join incentives B on A.EMPLOYEE\_ID=B.EMPLOYEE\_REF\_ID  
**SQL Queries in MySQL,** Select FIRST\_NAME, IFNULL(INCENTIVE\_AMOUNT,0) from employee a left join incentives B on A.EMPLOYEE\_ID=B.EMPLOYEE\_REF\_ID

**53. Select first\_name, incentive amount from employee and incentives table for all employees who got incentives using left join**

**SQL Queries in Oracle,** Select FIRST\_NAME,nvl(INCENTIVE\_AMOUNT,0) from employee a right join incentives B on A.EMPLOYEE\_ID=B.EMPLOYEE\_REF\_ID  
**SQL Queries in SQL Server,** Select FIRST\_NAME, isnull(INCENTIVE\_AMOUNT,0) from employee a right join incentives B on A.EMPLOYEE\_ID=B.EMPLOYEE\_REF\_ID  
**SQL Queries in MySQL,** Select FIRST\_NAME, IFNULL(INCENTIVE\_AMOUNT,0) from employee a right join incentives B on A.EMPLOYEE\_ID=B.EMPLOYEE\_REF\_ID

**54. Select max incentive with respect to employee from employee and incentives table using sub query**

**SQL Queries in Oracle,** select DEPARTMENT,(select nvl(max(INCENTIVE\_AMOUNT),0) from INCENTIVES where EMPLOYEE\_REF\_ID=EMPLOYEE\_ID) Max\_incentive from EMPLOYEE  
**SQL Queries in SQL Server,** select DEPARTMENT,(select ISNULL(max(INCENTIVE\_AMOUNT),0) from INCENTIVES where EMPLOYEE\_REF\_ID=EMPLOYEE\_ID) Max\_incentive from EMPLOYEE  
**SQL Queries in MYSQL Server,** select DEPARTMENT,(select IFNULL (max(INCENTIVE\_AMOUNT),0) from INCENTIVES where EMPLOYEE\_REF\_ID=EMPLOYEE\_ID) Max\_incentive from EMPLOYEE

# ****Top N Salary - Examples****

**55. Select TOP 2 salary from employee table**

**SQL Queries in Oracle,** select \* from (select \* from employee order by SALARY desc) where rownum <3  
**SQL Queries in SQL Server,** select top 2 \* from employee order by salary desc  
**SQL Queries in MySQL,** select \* from employee order by salary desc limit 2

**56. Select TOP N salary from employee table**

SQL Queries in Oracle, select \* from (select \* from employee order by SALARY desc) where rownum <N + 1  
SQL Queries in SQL Server, select top N \* from employee  
SQL Queries in MySQL, select \* from employee order by salary desc limit N

**57. Select 2nd Highest salary from employee table**

SQL Queries in Oracle, select min(salary) from (select \* from (select \* from employee order by SALARY desc) where rownum <3)  
SQL Queries in SQL Server, select min(SALARY) from (select top 2 \* from employee) a  
SQL Queries in MySQL, select min(SALARY) from (select \* from employee order by salary desc limit 2) a

**58. Select Nth Highest salary from employee table**

**SQL Queries in Oracle,** select min(salary) from (select \* from (select \* from employee order by SALARY desc) where rownum <N + 1)  
**SQL Queries in SQL Server,** select min(SALARY) from (select top N \* from employee) a  
**SQL Queries in MySQL,** select min(SALARY) from (select \* from employee order by salary desc limit N) a

# ****SQL Union - Examples****

**59. Select First\_Name,LAST\_NAME from employee table as separate rows**

select FIRST\_NAME from EMPLOYEE union select LAST\_NAME from EMPLOYEE

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

-UNION performs a DISTINCT on the result set, eliminating any duplicate rows.(remove intersection)

-UNION ALL does not remove duplicates, and it therefore faster than UNION.

**61. Select employee details from employee table if data exists in incentive table ?**

select \* from EMPLOYEE where exists (select \* from INCENTIVES)

**Explanation** : Here exists statement helps us to do the job of If statement. Main query will get executed if the sub query returns at least one row. So we can consider the sub query as "If condition" and the main query as "code block" inside the If condition. We can use any SQL commands (Joins, Group By , having etc) in sub query. This command will be useful in queries which need to detect an event and do some activity.

**62. How to fetch data that are common in two query results ?**

select \* from EMPLOYEE where EMPLOYEE\_ID INTERSECT select \* from EMPLOYEE where EMPLOYEE\_ID <4  
Explanation : Here INTERSECT command is used to fetch data that are common in 2 queries. In this example, we had taken EMPLOYEE table in both the queries.We can apply INTERSECT command on different tables. The result of the above query will return employee details of "ROY" because, employee id of ROY is 3, and both query results have the information about ROY.

**63. Get Employee ID's of those employees who didn't receive incentives without using sub query ?**

select EMPLOYEE\_ID from EMPLOYEE  
MINUS  
select EMPLOYEE\_REF\_ID from INCENTIVES  
 Explanation : To filter out certain information we use MINUS command. What MINUS Command odes is that, it returns all the results from the first query, that are not part of the second query. In our example, first three employees received the incentives. So query will return employee id's 4 to 8.

**64. Select 20 % of salary from John , 10% of Salary for Roy and for other 15 % of salary from employee table**

SELECT FIRST\_NAME, CASE FIRST\_NAME WHEN 'John' THEN SALARY \* .2 WHEN 'Roy' THEN SALARY \* .10 ELSE SALARY \* .15 END "Deduced\_Amount" FROM EMPLOYEE   
Explanation : Here we are using SQL CASE statement to achieve the desired results. After case statement, we had to specify the column on which filtering is applied. In our case it is "FIRST\_NAME". And in then condition, specify the name of filter like John, Roy etc. To handle conditions outside our filter, use else block where every one other than John and Roy enters.

**65. Select Banking as 'Bank Dept', Insurance as 'Insurance Dept' and Services as 'Services Dept' from employee table**

SQL Queries in Oracle, SELECT distinct DECODE (DEPARTMENT, 'Banking', 'Bank Dept', 'Insurance', 'Insurance Dept', 'Services', 'Services Dept') FROM EMPLOYEE  
SQL Queries in SQL Server and MySQL, SELECT case DEPARTMENT when 'Banking' then 'Bank Dept' when 'Insurance' then 'Insurance Dept' when 'Services' then 'Services Dept' end FROM EMPLOYEE  
  
**Explanation** : Here DECODE keyword is used to specify the alias name. In oracle we had specify, Column Name followed by Actual Name and Alias Name as arguments. In SQL Server and MySQL, we can use the earlier switch case statements for alias names.

**66. Delete employee data from employee table who got incentives in incentive table**

delete from EMPLOYEE where EMPLOYEE\_ID in (select EMPLOYEE\_REF\_ID from INCENTIVES)  
  
**Explanation** : Trick about this question is that we can't delete data from a table based on some condition in another table by joining them. Here to delete multiple entries from EMPLOYEE table, we need to use Subquery. Entries will get deleted based on the result of Subquery.

**67. Insert into employee table Last Name with " ' " (Single Quote - Special Character)**

Tip - Use another single quote before special character  
Insert into employee (LAST\_NAME) values ('Test''')

**68. Select Last Name from employee table which contain only numbers**

Select \* from EMPLOYEE where lower(LAST\_NAME)=upper(LAST\_NAME)  
**Explanation** : Here in order to achieve the desired result, we use ASCII property of the database. If we get results for a column using Lower and Upper commands, ASCII of both results will be same for numbers. If there is any alphabets in the column, results will differ.

**69. Write a query to rank employees based on their incentives for a month**

select FIRST\_NAME,INCENTIVE\_AMOUNT,DENSE\_RANK() OVER (PARTITION BY INCENTIVE\_DATE ORDER BY INCENTIVE\_AMOUNT DESC) AS Rank from EMPLOYEE a, INCENTIVES b where a.EMPLOYEE\_ID=b.EMPLOYEE\_REF\_ID  
**Explanation** : Here in order to rank employees based on their rank for a month, DENSE\_RANK keyword is used. Here partition by keyword helps us to sort the column with which filtering is done. Rank is provided to the column specified in the order by statement. The above query ranks employees with respect to their incentives for a given month.

**70. Update incentive table where employee name is 'John'**

**Explanation** : Here we need to join Employee and Incentive Table for updating the incentive amount. But for update statement joining query wont work. We need to use sub query to update the data in the incentive table. SQL Query is as shown below.

update INCENTIVES set INCENTIVE\_AMOUNT='9000' where EMPLOYEE\_REF\_ID=(select EMPLOYEE\_ID from EMPLOYEE where FIRST\_NAME='John' )

# Self Join

SELECT

    (e.first\_name || '  ' || e.last\_name) employee,

    (m.first\_name || '  ' || m.last\_name) manager,

    e.job\_title

FROM

    employees e

LEFT JOIN employees m ON

    m.employee\_id = e.manager\_id

ORDER BY

    manager;

# ****SQL Table Scripts - Examples****

**71. Write create table syntax for employee table**

Oracle –

CREATE TABLE EMPLOYEE (  
EMPLOYEE\_ID NUMBER,  
FIRST\_NAME VARCHAR2(20 BYTE),  
LAST\_NAME VARCHAR2(20 BYTE),  
SALARY FLOAT(126),  
JOINING\_DATE TIMESTAMP (6) DEFAULT sysdate,  
DEPARTMENT VARCHAR2(30 BYTE) )

**72. Write syntax to delete table employee**

DROP table employee;

**73. Write syntax to set EMPLOYEE\_ID as primary key in employee table**

ALTER TABLE EMPLOYEE add CONSTRAINT EMPLOYEE\_PK PRIMARY KEY(EMPLOYEE\_ID)

**74. Write syntax to set 2 fields(EMPLOYEE\_ID,FIRST\_NAME) as primary key in employee table**

ALTER TABLE EMPLOYEE add CONSTRAINT EMPLOYEE\_PK PRIMARY KEY(EMPLOYEE\_ID,FIRST\_NAME)

**75. Write syntax to drop primary key on employee table**

Alter TABLE EMPLOYEE drop CONSTRAINT EMPLOYEE\_PK;

**76. Write Sql Syntax to create EMPLOYEE\_REF\_ID in INCENTIVES table as foreign key with respect to EMPLOYEE\_ID in employee table**

ALTER TABLE INCENTIVES ADD CONSTRAINT INCENTIVES\_FK FOREIGN KEY (EMPLOYEE\_REF\_ID) REFERENCES EMPLOYEE(EMPLOYEE\_ID)

**77. Write SQL to drop foreign key on employee table**

ALTER TABLE INCENTIVES drop CONSTRAINT INCENTIVES\_FK;

**78. Write SQL to create Oracle Sequence**

Use the CREATE SEQUENCE statement to create a **sequence**, which is a database object from which multiple users may generate unique integers. You can use sequences to automatically generate primary key values.

CREATE SEQUENCE EMPLOYEE\_ID\_SEQ

START WITH 0

INCREMENT BY1

NOMAXVALUE

MINVALUE 0

NOCYCLE

NOCACHE

NOORDER;

**78. Write SQL to create Oracle Procedure**

**Oracle Procedure** is block which performs one or more specific task.

CREATE or replace PROCEDURE remove\_emp (employee\_id NUMBER)

AS

*tot\_emps NUMBER;*

*BEGIN*

*DELETE FROM employees*

*WHERE employees.employee\_id = remove\_emp.employee\_id;*

*tot\_emps := tot\_emps - 1;*

*END*;

**78. Write SQL to create Oracle Function**

CREATE or replace FUNCTION get\_bal(acc\_no IN NUMBER)

RETURN NUMBER

IS

acc\_bal NUMBER(11,2);

*BEGIN*

*SELECT order\_total*

*INTO acc\_bal*

*FROM orders*

*WHERE customer\_id = acc\_no;*

*RETURN(acc\_bal);*

*END;*

**79. Write Sql syntax to create Oracle Trigger before insert of each row in employee table**

CREATE[OR REPLACE ]TRIGGER trigger\_name

{BEFORE | AFTER | INSTEAD OF}

{INSERT[OR]|UPDATE[OR]|DELETE}

[OF col\_name]

ON table\_name

[REFERENCING OLD AS o NEW AS n]

[FOR EACH ROW]

WHEN(condition)

DECLARE

Declaration-statements

BEGIN

Executable-statements

EXCEPTION

Exception-handling-statements

END;

CREATE OR REPLACE TRIGGER EMPLOYEE\_ROW\_ID\_TRIGGER  
BEFORE INSERT ON EMPLOYEE FOR EACH ROW  
DECLARE  
seq\_no number(12);  
BEGIN  
select EMPLOYEE\_ID\_SEQ.nextval into seq\_no from dual ;  
:new EMPLOYEE\_ID :=seq\_no;  
END;  
SHOW ERRORS;

**80. Oracle Procedure81. Oracle View**

An example oracle view script is given below  
create view Employee\_Incentive as select FIRST\_NAME,max(INCENTIVE\_AMOUNT) INCENTIVE\_AMOUNT from EMPLOYEE a, INCENTIVES b where a.EMPLOYEE\_ID=b.EMPLOYEE\_REF\_ID group by FIRST\_NAME

**82. Oracle materialized view - Daily Auto Refresh**

CREATE MATERIALIZED VIEW Employee\_Incentive  
REFRESH COMPLETE  
START WITH SYSDATE  
NEXT SYSDATE + 1 AS  
select FIRST\_NAME,INCENTIVE\_DATE,INCENTIVE\_AMOUNT from EMPLOYEE a, INCENTIVES b   
where a.EMPLOYEE\_ID=b.EMPLOYEE\_REF\_ID

**83. Oracle materialized view - Fast Refresh on Commit**

Create materialized view log for fast refresh. Following materialized view script wont get executed if materialized view log doesn't exists  
  
CREATE MATERIALIZED VIEW MAT\_Employee\_Incentive\_Refresh  
BUILD IMMEDIATE  
REFRESH FAST ON COMMIT AS  
select FIRST\_NAME,max(INCENTIVE\_AMOUNT) from EMPLOYEE a, INCENTIVES b  
where a.EMPLOYEE\_ID=b.EMPLOYEE\_REF\_ID group by FIRST\_NAME

**84. What is SQL Injection ?**

SQL Injection is one of the techniques uses by hackers to hack a website by injecting SQL commands in data fields.

**85. What’s the Difference between a Primary Key and a Unique Key?**

Both primary key and unique key enforce uniqueness of the column on which they are defined. But by default, the primary key creates a clustered index on the column, whereas unique key creates a non-clustered index by default. Another major difference is that primary key doesn’t allow NULLs, but unique key allows one NULL only.