|  |  |
| --- | --- |
| **Topic** | Oracle SQL Language Fundamentals I |
| **Document Name** | SQL03-EX-01-05 |
|  |  |

## Exercise SQL03-EX-01:

**Definiton :** Write followig SQL queries:

* Add a colum to employees table named MAX\_SALARY.
* Update MAX\_SALARY with maximum salary amount with subquery.
* Delete employee who have minimum salary using subquery.

**SQL:**

ALTER TABLE hr.employees

ADD max\_salary number(10);

UPDATE hr.employees

SET max\_salary = (SELECT max(salary)

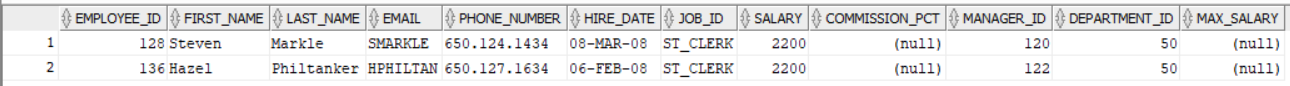
FROM hr.employees);

DELETE from hr.employees

WHERE salary = (select min(salary) from hr.employees);

**Screenshot:**





## Exercise SQL03-EX-02:

**Definiton :** Write followig SQL queries:

* Define index (named DPR\_NAME\_IDX) on DEPARTMENT\_NAME column of DEPARTMENTS table.
* Define constraint (named CNSTR\_SALARY) on employee salary. (Salary must be between 1000$ and 100.000$)
* Drop defined index.
* Enable, disable, drop defined constraint.

**SQL:**

CREATE INDEX DPR\_NAME\_IDX

ON hr.departments(DEPARTMENT\_NAME);

alter table hr.employees

add constraint CNSTR\_SALARY check (salary >= 1000 and salary <= 100000);

DROP INDEX DPR\_NAME\_IDX;

ALTER TABLE hr.employees

DISABLE CONSTRAINT CNSTR\_SALARY;

ALTER TABLE hr.employees

ENABLE CONSTRAINT CNSTR\_SALARY;

ALTER TABLE hr.employees

DROP CONSTRAINT CNSTR\_SALARY;

## Exercise SQL03-EX-03:

**Definiton :** Create a table from EMPLOYEES with distinct department\_id column. Add department\_name to that table. With DEPARTMENTS table, update department\_name for included department\_ids and insert department\_id and department\_name values for not included rows. Use MERGE keyword.

**SQL:**

CREATE TABLE temp

AS (SELECT DISTINCT department\_id

FROM hr.employees);

ALTER TABLE temp

ADD department\_name varchar(20);

MERGE INTO temp x

USING (SELECT department\_id, department\_name FROM hr.departments) y

ON (x.department\_id = y.department\_id)

WHEN MATCHED THEN

UPDATE SET x.department\_name = y.department\_name

WHEN NOT MATCHED THEN

INSERT(x.department\_id, x.department\_name)

VALUES(y.department\_id, y.department\_name);

**Screenshot:**

## Exercise SQL03-EX-04:

**Definiton :** Using **WITH** keyword, do following jobs:

* Firstly select first\_name, last\_name, job\_id, department\_id from employees table whoes job\_id starts with ‘S’.
* Additionally select job\_title and min-max salary amount.
* Add department\_name to that query.
* Lastly concat first\_name and last\_name with space as full\_name alias and list with other selected columns.

**SQL:**

WITH EmpJobStartingWithS AS (

SELECT (first\_name || ' ' || last\_name) as full\_name, job\_id, department\_id

FROM hr.employees

WHERE job\_id LIKE 'S%'

)

SELECT a.full\_name, a.job\_id, a.department\_id,

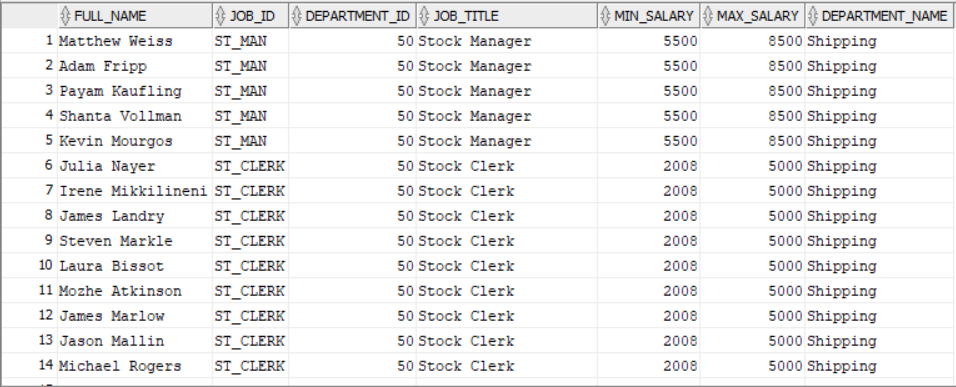
m.job\_title, m.min\_salary, m.max\_salary, x.department\_name

FROM EmpJobStartingWithS a

INNER JOIN hr.jobs m ON a.job\_id = m.job\_id

INNER JOIN hr.departments x ON a.department\_id = x.department\_id

**Screenshot:**



## Exercise SQL03-EX-05:

**Definiton :** Search for COMMIT and ROLLBACK keywords and explain them.

COMMIT argument pinpoints the place where it is written. When we use ROLLBACK we can get back to the last committed pinpoint.