CSE 311L(Database Management System)

LAB-Week 07 (Part A)

Manipulating Data

Topics:

- Copying Rows from Another Table
- Updating Rows in a Table
- Updating Rows Based on Another Table
- Example of Merging Rows

Copying Rows from Another Table

```
INSERT INTO sales_reps(id, name, salary, commission_pct)
     SELECT employee_id, last_name, salary, commission_pct
     FROM employees
     WHERE job_id LIKE '%REP%';
```

Updating Rows in a Table

```
     UPDATE employees
     SET department_id = 70
     WHERE employee_id = 113;
```

Updating Rows Based on Another Table

```
     UPDATE copy_emp
     SET department_id =(SELECT department_id
                         FROM employees
                         WHERE employee_id = 100)
     WHERE job_id      = (SELECT job_id
                         FROM employees
                         WHERE employee_id = 200);
```

Example of Merging Rows

```
MERGE INTO copy_emp c
     USING employees e
     ON (c.employee_id = e.employee_id)
WHEN MATCHED THEN
   UPDATE SET
     c.first_name = e.first_name,
     c.last_name = e.last_name,
     c.email = e.email,
     c.phone_number = e.phone_number,
     c.hire_date = e.hire_date,
     c.job_id = e.job_id,
     c.salary = e.salary,
     c.commission_pct = e.commission_pct,
     c.manager_id = e.manager_id,
     c.department_id = e.department_id
WHEN NOT MATCHED THEN
  INSERT VALUES(e.employee_id, e.first_name, e.last_name,
     e.email, e.phone_number, e.hire_date, e.job_id,
     e.salary, e.commission_pct, e.manager_id,
     e.department_id);
```

Activity 01:

Create a table that has some fields similar to employees table. Then insert 5 rows to the new table. Afterwards, merge the new table and the employees table to a new table based on employee number.

CSE 311L(Database Management System)

LAB-Week 07 (Part B)

Managing Tables

Topics:

- The ALTER TABLE Statement
- Adding a Column
- Modifying a Column
- Dropping a Column
- Changing the Name of an Object
- Truncating a Table
- Add PRIMARY KEY/ FOREIGN KEY constraints
- CREATE VIEW

The ALTER TABLE Statement

Use the `ALTER TABLE` statement to:

- Add a new column
- Modify an existing column
- Define a default value for the new column
- Drop a column

Adding a Column

```
ALTER TABLE dept80
ADD (job_id VARCHAR2(9));
```

Modifying a Column

```
ALTER TABLE dept80
MODIFY (last_name VARCHAR2(30));
```

Dropping a Column

```
ALTER TABLE dept80
DROP COLUMN job_id;
```

Dropping a Table

```
DROP TABLE dept80;
```

Changing the Name of an Object

```
RENAME dept TO detail_dept;
```

Truncating a Table

```
TRUNCATE TABLE detail_dept;
```

Add PRIMARY KEY/ FOREIGN KEY constraints

```
ALTER TABLE employees
ADD CONSTRAINT emp_manager_fk
FOREIGN KEY(manager_id)
REFERENCES employees(employee_id);
```

Creating a View

- **Create a view by using column aliases in the subquery.**

```
CREATE VIEW  salvu50
AS SELECT  employee_id ID_NUMBER, last_name NAME,
           salary*12 ANN_SALARY
   FROM    employees
   WHERE   department_id = 50;
View created.
```

Activity 01:

Create the EMP table based on the following table instance chart..

Name	Null?	Type
ID		NUMBER(7)
LAST_NAME		VARCHAR2(25)
FIRST_NAME		VARCHAR2(25)
DEPT_ID		NUMBER(7)

a. Modify the EMP table to allow for longer employee last names. Confirm your modification.

b. Create the EMPLOYEES2 table based on the structure of the EMPLOYEES table. Include only the EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY, and DEPARTMENT_ID columns. Name the columns in your new table ID, FIRST_NAME, LAST_NAME, SALARY , and DEPT_ID, respectively.

c. Drop the EMP table.

d. Rename the EMPLOYEES2 table as EMP.

e. Drop the FIRST_NAME column from the EMP table. Confirm your modification by checking the description of the table.