CSE 311L(Database Management System)

LAB-Week 07 (Lecture 1)

Manipulating Data

Topics:

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

Copy table structure from another table

```
CREATE table employees_copy LIKE employees;
```

Copy table structure and data from another table

```
CREATE TABLE emp_cpy_1 SELECT * from employees
```

Copying Rows from Another Table

```
INSERT INTO emp_cpy (Employee_Id, Department_id, Commission_pct,
Last_Name, Email, Hire_Date, Job_Id)
SELECT Employee_Id, Department_id, Commission_pct, Last_Name, Email,
Hire_Date, Job_Id
FROM employees
WHERE Job_Id LIKE "%REP%"
```

Updating Rows in a Table

```
UPDATE emp_cpy_1
SET Department_id = 70
WHERE Employee_Id = 113
```

Updating Rows Based on Another Table

```
UPDATE emp_cpy_1
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

- **IGNORE** keyword allows those rows in emp_cpy to supersede those in employees that have a matching primary key, while still inserting rows with new primary keys.
- **REPLACE** keyword will update those rows already in emp_cpy with the corresponding row from employees, while inserting rows with new primary keys.

```
INSERT IGNORE
INTO emp_cpy
SELECT *
FROM employees

REPLACE
INTO emp_cpy
SELECT * FROM employees
```

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 (Lecture 2)

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 employees
ADD COLUMN job_id_1 VARCHAR(90)
```

Modifying a Column

```
ALTER TABLE employees
MODIFY  COLUMN job_id_1 varchar(100)
```

Dropping a Column

```
ALTER TABLE employees
DROP COLUMN job_id_1
```

Dropping a Table

```
DROP TABLE emp_cpy_1
```

Changing the Name of an Object

```
RENAME TABLE emp_cpy TO emp_cpy_1;
```

Truncating a Table

```
TRUNCATE TABLE emp_cpy_1;
```

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 view EMPLOYEE_MANAGER_INFORMATION AS
SELECT worker.First_Name "EMP_First_Name", worker.Manager_id, manager.First_Name
AS "MGR_First_Name"
FROM employees worker
JOIN employees manager
ON(worker.Manager_id = manager.Employee_Id)
```

Query the View

```
SELECT * FROM `employee_manager_information`
```

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.