BIC 21404 DATABASE SYSTEM

LAB SHEET 2

Title	: Creating and managing tables
Objectives	: At the end of the session, students are able to:
	i. Create table using command line
	ii. Alter tables
	iii. Inserting data into tables
Duration	: 2 Hours

Introduction to Schema Objects

A database schema is a logical container for data structures, called schema objects. Examples of schema objects are tables and indexes. Schema objects are created and manipulated with SQL.

A database user has a password and various database privileges. Each user owns a single schema, which has the same name as the user. The schema contains the data for the user owning the schema. For example, the hr user owns the hr schema, which contains schema objects such as the employees table. In a production database, the schema owner usually represents a database application rather than a person.

Within a schema, each schema object of a particular type has a unique name. For example, hr.employees refers to the table employees in the hr schema. Figure 2-1 depicts a schema owner named hr and schema objects within the hr schema.

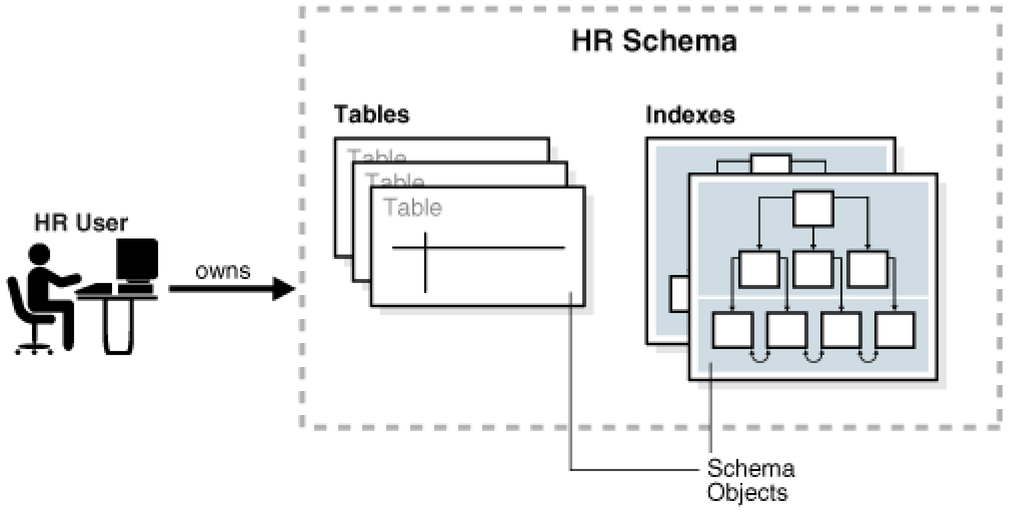

Figure 2-1: HR Schema

The hr schema is a sample schema that contains information about employees, departments and locations, work histories, and so on. The following Figure 2-2 is an entity-relationship diagram of the tables in the hr schema.

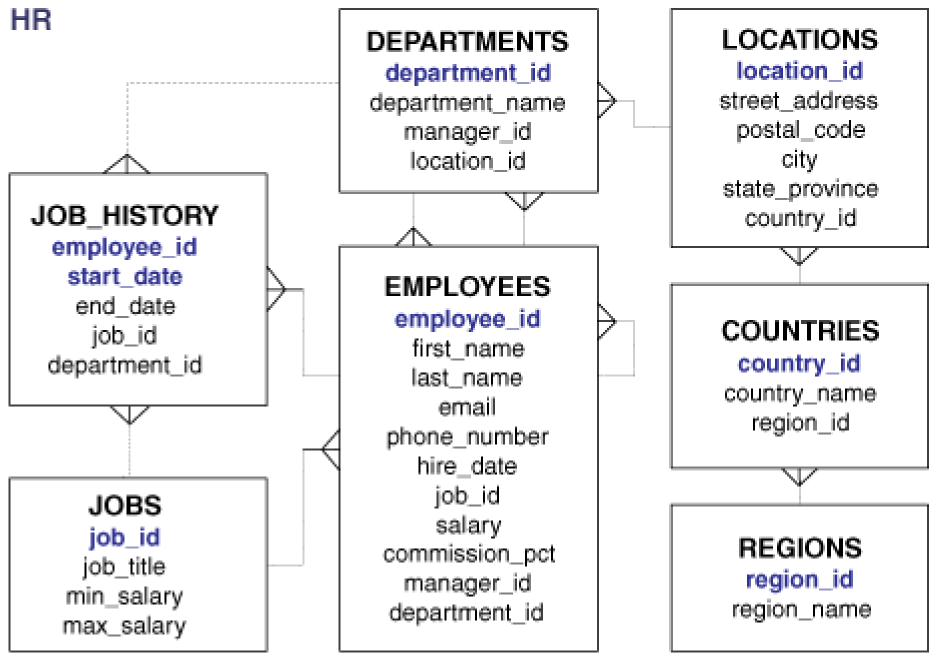

Figure 2-2: HR Schema

Exercise 1: Creating a Database User

You must create at least one database user that you will use to create database objects. A database user is a type of database object: a user is associated with a database schema, you connect to the database as a database user, and the database user is the owner of any database objects (tables and so on) that you create in the schema associated with the user.

For example, to create a database user named hr. Follow these steps, using the command line:

1. Display the SQL command prompt window. For example, on Windows, click Start, then Programs (or All Programs), then Oracle Database 11g Express Edition, and then Run SQL Command Line.

2. Connect as the SYSTEM user:
   - Type: connect
   - Enter user-name: system
   - Enter password: <system-password>

3. Create the user. For example, enter a statement in the following form:
```
SQL> create user hr identified by <password-for-hr>;
```

4. Grant the user the necessary privileges. For example:
```
SQL> grant connect, resource to hr;
```

5. Optionally, exit SQL*Plus (which also closes the command window):
```
SQL> exit
```

Overview of Tables

A table is the basic unit of data organization in an Oracle database. A table describes an entity, which is something of significance about which information must be recorded. For example, an employee could be an entity.

A table definition includes a table name and set of columns. A column identifies an attribute of the entity described by the table. For example, the column `employee_id` in the employees table refers to the employee ID attribute of an employee entity.

In general, you give each column a column name, a data type, and a width when you create a table. For example, the data type for employee_id is `NUMBER(6)`, indicating that this column can only contain numeric data up to 6 digits in width. The width can be predetermined by the data type, as in the case of `DATE`.

A table can contain a virtual column, which unlike a nonvirtual column does not consume disk space. The database derives the values in a virtual column on demand by computing a set of user-specified expressions or functions. For example, the virtual column income could be a function of the `salary` and `commission_pct` columns.

After you create a table, you can insert, query, delete, and update rows using SQL. A row is a collection of column information corresponding to a record in a table. For example, a row in the employees table describes the attributes of a specific employee.

HR Table Descriptions

Table DEPARTMENTS

```
Name                                      Null?    Type
----------------------------------------- -------- --------------------------
DEPARTMENT_ID                             NOT NULL NUMBER(4)
DEPARTMENT_NAME                           NOT NULL VARCHAR2(30)
MANAGER_ID                                         NUMBER(6)
LOCATION_ID                                        NUMBER(4)
```

Table EMPLOYEES

```
Name                                      Null?    Type
----------------------------------------- -------- --------------------------
EMPLOYEE_ID                               NOT NULL NUMBER(6)
FIRST_NAME                                         VARCHAR2(20)
LAST_NAME                                 NOT NULL VARCHAR2(25)
EMAIL                                     NOT NULL VARCHAR2(25)
PHONE_NUMBER                                       VARCHAR2(20)
HIRE_DATE                                 NOT NULL DATE
JOB_ID                                    NOT NULL VARCHAR2(10)
SALARY                                             NUMBER(8,2)
COMMISSION_PCT                                     NUMBER(2,2)
MANAGER_ID                                         NUMBER(6)
DEPARTMENT_ID                                      NUMBER(4)
```

Table JOBS

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
JOB_ID                                    NOT NULL VARCHAR2(10)
JOB_TITLE                                 NOT NULL VARCHAR2(35)
MIN_SALARY                                         NUMBER(6)
MAX_SALARY                                         NUMBER(6)
```

Table JOB_HISTORY

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
EMPLOYEE_ID                               NOT NULL NUMBER(6)
START_DATE                                NOT NULL DATE
END_DATE                                  NOT NULL DATE
JOB_ID                                    NOT NULL VARCHAR2(10)
DEPARTMENT_ID                                      NUMBER(4)
```

Table LOCATIONS

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
LOCATION_ID                               NOT NULL NUMBER(4)
STREET_ADDRESS                                     VARCHAR2(40)
POSTAL_CODE                                        VARCHAR2(12)
CITY                                      NOT NULL VARCHAR2(30)
STATE_PROVINCE                                     VARCHAR2(25)
COUNTRY_ID                                         CHAR(2)
```

Table REGIONS

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
REGION_ID                                 NOT NULL NUMBER
REGION_NAME                                        VARCHAR2(25)
```

Table COUNTRIES

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
COUNTRY_ID                                NOT NULL CHAR(2)
COUNTRY_NAME                                       VARCHAR2(40)
REGION_ID                                          NUMBER
```

Exercise 2: Creating Tables

1. Run SQL Command Line.
2. Connect as the hr user:
```
SQL> connect hr/<password-for-hr>;
```
3. Create table `employees`:
```
CREATE TABLE employees (
   employee_id    NUMBER(6),
   first_name     VARCHAR2(20),
   last_name      VARCHAR2(25)  CONSTRAINT emp_lname_nn  NOT NULL,
   email          VARCHAR2(25)  CONSTRAINT emp_email_nn  NOT NULL,
   phone_number   VARCHAR2(20),
   hire_date      DATE          CONSTRAINT emp_hdate_nn  NOT NULL,
   job_id         VARCHAR2(10)  CONSTRAINT emp_job_nn  NOT NULL,
   salary         NUMBER(8,2),
   commission_pct NUMBER(2,2),
   manager_id     NUMBER(6),
   department_id  NUMBER(4),
   CONSTRAINT     emp_salary_min CHECK (salary > 0),
   CONSTRAINT     emp_email_uk   UNIQUE (email)
) ;
```
4. Create all HR tables by referring the HR schema diagram in Figure 2-2 and the HR Table Description.

Exercise 3: Displaying Table Structure

Use the `DESCRIBE` command to display the structure of a table. The command displays the column names and the data types, and it shows you whether a column must contain data (that is whether the column has a `NOT NULL` constraint).

1. Type the following command
```
DESCRIBE employees;
```
2. Type the following command for each of tables created earlier.

Exercise 4: Alter Tables

1. Run SQL Command Line.
2. Connect as the hr user: `SQL> connect hr/<password-for-hr>;`
3. ALTER TABLE statement that adds integrity constraints to the employees table. Integrity constraints enforce business rules and prevent the entry of invalid information into tables.
```
ALTER TABLE employees
ADD (CONSTRAINT emp_emp_id_pk  PRIMARY KEY (employee_id),
     CONSTRAINT emp_dept_fk    FOREIGN KEY (department_id)
          REFERENCES departments,
     CONSTRAINT emp_job_fk     FOREIGN KEY (job_id)
          REFERENCES jobs (job_id),
     CONSTRAINT emp_manager_fk  FOREIGN KEY (manager_id)
          REFERENCES employees
) ;
```
4. Alter all HR tables to adds integrity constraints by referring the HR schema diagram in Figure 2-2.

Exercise 4: Insert Data Into Tables

1. Use `INSERT INTO` command to insert data into tables.

```
INSERT INTO employees
   (employee_id, first_name, last_name, email, phone_number, hire_date,
   job_id, salary, commission_pct, manager_id,department_id)
VALUES
   (100,'Steven',  'King',  'sking@hotmail.com','07-4533112','22-JAN-00',
   'SA_REP', 2400, 0.9, 200, 8);
```

2. Use insert command to insert sample data for all table that you have created.