**Creating Database and Table**

**Objectives**

* Set up database and schema;
* Create table and define its structure;
* Set constraints on columns;
* Make a sequence;
* Insert data to the table;
* Display rows of data using SELECT statement;
* Alter structure of the table;
* Delete rows of data;
* Update rows using SET statement;

Database may contain objects such as schema, tables, views, and application objects (sequences, stored procedures, user defined functions, and triggers) to name a few. We may create same nameof tables or views but they are instore in different schema. Schemas are namespaces that can be contained on the same database.

One of the objectives of this activity is to set a database and create tables. Below is the entity relationship diagram (ERD) of the tables we will create on a database. The name of the company that we will set up a database is Hope, Incorporated.

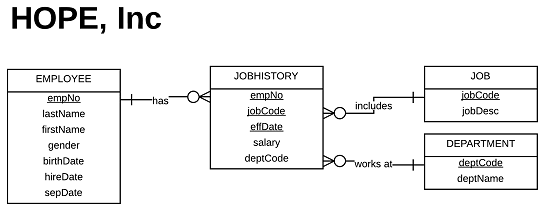


Figure 2-1. ERD of Hope, Inc.

Let us launch our DB2 Command Line Processor. Click Start | All Programs | DB2 | DB2COPY1(Default) | Command Line Processor.

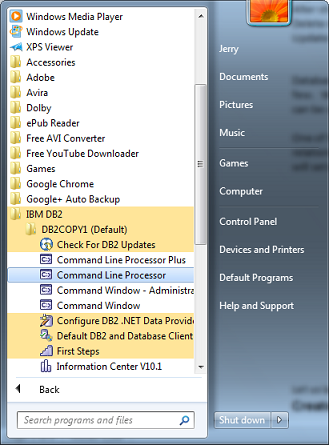


Figure 2-2. Launching Command Line Processor

**Create Database**

To create database, we use the following syntax:

CREATE DATABASE <database name>

Let us create database with database name **hope**.

CREATE DATABASE hope

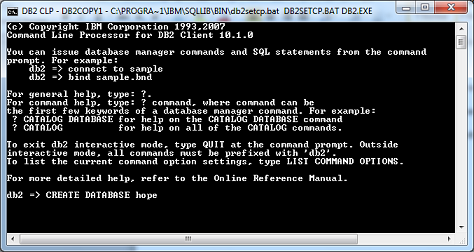


Figure 2-3. Creating database hope

**Open database**

To connect to an existing database we use the following syntax:

CONNECT TO <database name>

Let us open our newly created database. Issue the statement after creating the database.

CONNECT TO hope

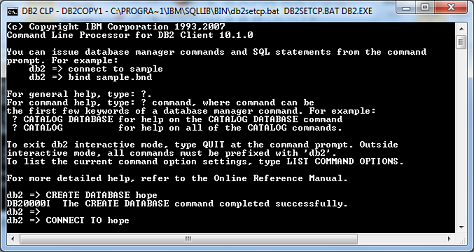


Figure 2-4. Opening database hope

You will notice upon connection to the database, the current version of DB2 installed is listed with the user id jerry and the database name hope. Any statement that contains no error has a notification command completed successfully.

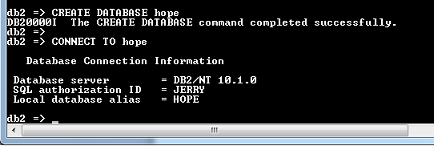
****

Figure 2-5. User Id and name of database

**Create Schema**

As mentioned earlier, schemas are namespaces. A database may contain one or more schemas.The default schema name is the user id. But we can create our own schema with the following syntax:

CREATE SCHEMA <schema name> AUTHORIZATION <user id>

Creating schema means you can create same table name but with different schema. Schema is one of the database objects.

Let us create schema with the name store.

CREATE SCHEMA store AUTHORIZATION jerry

You may drop the AUTHORIZATION syntax if you own or the one using the database. The default authorization is the user id. You may simply state:

CREATE SCHEMA store

To check if your schema is successfully created and logged, type the following script:

SELECT SCHEMANAME FROM SYSCAT.SCHEMATA

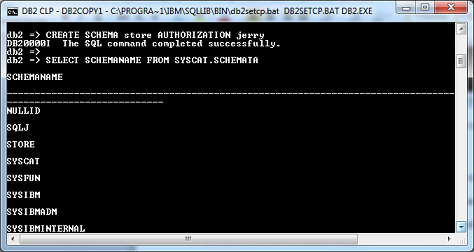


Figure 2-6. Creating schema and listing it.

Above you could see the STORE is listed under SCHEMANAME column from SCHEMATA table.

**Create Table**

We will now create table under our newly created schema. Table consists of rows and columns arranged logically. This is where we store data.

Columns or fields contain data with the same data type. Column data type is used to set size; hold a numeric value, text, or combination of both; store big images such as pictures and videos; specify date and time; restrict certain values it may contain. Examples of data types are: VARCHAR, CHAR, DECIMAL, INTEGER, DATE, CLOB, and BLOB.

Rows contain set values for each column. It may pertain that row is set of records at a particular instance in a table.

Think rows and columns as a matrix where a particular data are in stored.

Table DEPARTMENT will be composed of deptCode and deptName columns. **deptCode** will be set as primary key. Type:

CREATE TABLE store.department (deptCode VARCHAR(3) NOT NULL,

deptName VARCHAR (20),

PRIMARY KEY (deptCode))

Here, we explicitly identified the schema name of on where department table will be created. VARCHAR (3) of deptCode and VARCHAR(20) of deptName are variable data type with the size of 3 and 20 characters. Columns set with **VARCHAR** data type can contain combination of letters, numbers, and symbols.

We set **deptCode** as **primary key**. Primary key should not contain null value that is why we set the **NOT NULL** (unknown state) expression. Note that when column/s is/are set as primary key, no same value shall exist when inserting a record with the specific column/s. Primary key sets uniqueness of row in the entire table.



Figure 2-7. Creating table department

**Display the table structure**

To check the composition of our table, we issue a script

DESCRIBE TABLE store.department

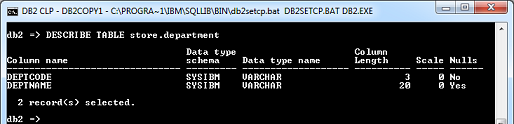


Figure 2-8. Check structure of the table using DESCRIBE TABLE

DB2 is not case-sensitive. Commands or statement issued and column names set are all considered in upper case letter. Lowering of letter cases in this book are made to give emphasis and differentiate user supplied names with DB2 statements.

Note: To adjust command line window display, right click its title. Click Properties | Layout. Set Screen Buffer Size width to 100 and height to 300.

**Insert data to table**

Use the following syntax:

INSERT INTO <schema name>.<table name> (<column name 1>, < column name 2>, …

column name n)

VALUES ( <value for column name 1>, (value for column name2, …

<value for column name n>)

Hence, we insert our first row:

INSERT INTO store.department (deptCode, deptName)

VALUES (‘IT’,’Information Tech’)

Enclose with single quotes data with VARCHAR data type. Separate column from another column with comma.

**Display row inserted**

To display the record on a table, we have the following syntax:

SELECT <column name 1>, <column name 2>, … <column name n>

FROM <schema name>. <table name>

List the content of department table. Issue the script below:

SELECT deptCode, deptName FROM store.department

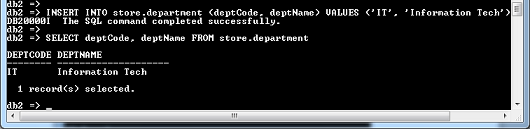


Figure 2-9. Displaying the rows of the table

We may use also asterisk symbol to display all columns of the table.



Figure 2-10. Use \* to display all columns

**Set the default schema**

To avoid retyping the schema name of a specific table, we set the schema we will use. On our case we will use store:

SET SCHEMA <schema name>

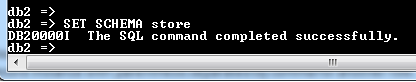


Figure 2-11. Setting the default schema

If you did not set the schema as you start with the Command Line Processor, the default schema is the user id. In our case the user id is jerry.

Insert again the following rows to department table:

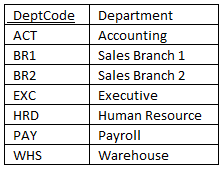


Table 2.1 Departments to insert

We use the following script:

INSERT INTO department VALUES (‘ACT’, ‘Accounting’),

(‘BR1’, ‘Sales Branch 1’),

(‘BR2’, ‘Sales Branch 2’),

(‘EXC’, ‘Executive’),

(‘HRD’, ‘Human Resource’),

(‘PAY’, ‘Payroll’),

(‘WHS’, ‘Warehouse’)

Notice that we did not specify anymore the schema name because it is understood that the schema to use is store. Another variety in inserting rows is dropping column names and typing the record instance after the VALUES expression. Group record instance with parenthesis separated by comma for another set of row.

Display the content of the department table.

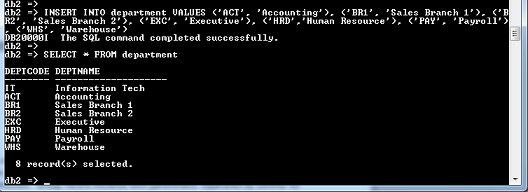


Figure 2-12. Inserting rows and displaying table contents

**Set primary key and check constraints**

Constraints are used to limit or validate data that will be inserted to the table maintained. As defined earlier, primary key is used to set uniqueness of the row from another row instance. This ensures that no other records or rows will duplicate the values once it is already used or set.

Check constraint is used to set rules on what values are accepted on that particular column. Insertion of row will not be possible unless validated by check constraint rule.

Let us create **employee** table with the following properties:

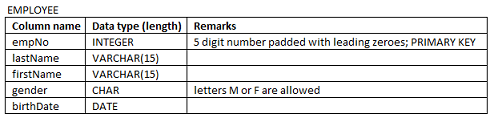


Table 2.2 Employee table structure

We type:

CREATE TABLE employee (**empNo** VARCHAR(5) NOT NULL PRIMARY KEY,

**lastName** VARCHAR(15),

**firstName** VARCHAR(15),

**gender** CHAR(1) CONSTRAINT **gender**\_**ck** CHECK (**gender** in (‘M’,’F’)),

**birthDate** DATE)



Figure 2-13. Creating table with constraints

We define gender\_ck as our constraint name for the column gender which has CHAR data type. The column accepts only values M and F.

birthDate is set with DATE data type.

Duplication of empNo is not allowed because this is set as PRIMARY KEY. As discussed, primary key should not contain NULL value.

As we describe our table we have:



Figure 2-14. Employee table structure

**Create a Sequence**

As indicated on our requirement setting, employee number should be numbers five in length where the remaining leading numbers are padded with zeroes. Assume also that empNo are in odd numbers. Here we will create a sequence key to set number automatically in odd number:

CREATE SEQUENCE empNo\_seq

START WITH 1

INCREMENT BY 2

NO CYCLE

CACHE 5

empNo\_seq generates number starting with 1 and next number will increment by 2. CACHE 5 is to store next 5 numbers in the memory. This mean that numbers 1, 3, 5, 7, 9 are already reserved in case connection during insertion is interrupted.

To facilitate the use of sequence, we will use NEXT VALUE expression to return the sequence value on our INSERT command:

INSERT INTO employee VALUES (LPAD(NEXT VALUE FOR empNo\_seq, 5,’0’), ‘Smith’, ‘John’,

‘M’, ‘1985-01-30’)

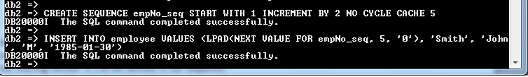


Figure 2-15. Using Sequence and LPAD() function

When you display all records we will have this:



Figure 2-16. Current content of employee table

We used LPAD() function to pad zeroes to our sequence number. 5 means the size of the column and 0 is the character that will be padded starting from left. NEXT VALUE FOR empNo\_seq contains 1 since our starting number is 1 as we defined in our sequence.

Type the following insertion of rows to check if our sequence is working:

INSERT INTO employee VALUES (LPAD(NEXT VALUE FOR empNo\_seq, 5,’0’), ‘Smith’, ‘Jane’,

‘F’, ‘1990-05-16’),

(LPAD(NEXT VALUE FOR empNo\_seq, 5,’0’), ‘King’, ‘Don’,

‘M’, ‘1986-02-14’)

Display all records to check if it is working. You should have the following output:

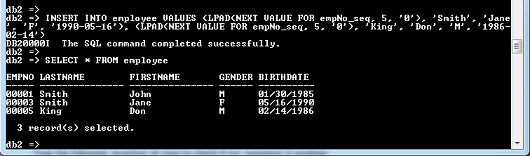


Figure 2-17. Current content of employees table after insertion

Terminate DB2 and reconnect again to HOPE database. After reconnection we set it to our STORE schema and insert a new set of row:

TERMINATE

DB2

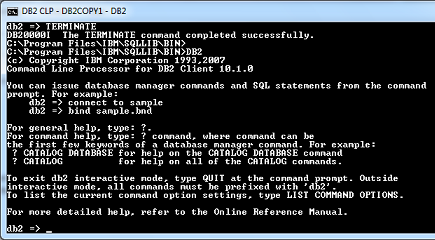


Figure 2-18. Terminating and reconnecting DB2 connection

CONNECT TO hope

SET SCHEMA store

INSERT INTO employee VALUES (LPAD(NEXT VALUE FOR empNo\_seq, 5,’0’), ‘Davis’, ‘Tom’,

‘M’, ‘1989-12-16’)

SELECT \* FROM employee

The last part should be the output:

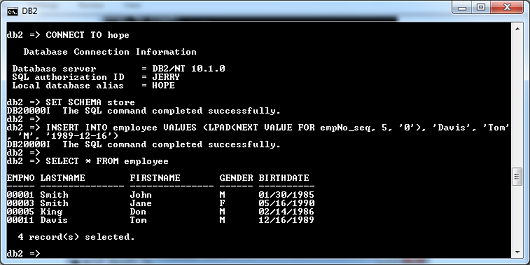


Figure 2-19. Changes in the sequence number

The question is: Why empNo is set to 11 and not 7?

When we terminate the connection other numbers such as 7 and 9 were no longer effective since these numbers were part of our cache which was set to 5.

Therefore, the next value to generate upon re-connection is 11.

**Create composite and foreign keys**

We continue to implement the ERD of Hope, Inc. Next to our list is to create job and jobHistory tables. Let us start with job table first.

CREATE TABLE job (jobCode VARCHAR(4) NOT NULL PRIMARY KEY, jobDesc VARCHAR(20))



Figure 2-20. Job table created

**Using IMPORT statement**

Instead of issuing series of INSERT INTO statements, let us use IMPORT statement to retrieve DELIMITED CSV file as a source of our data. Entering and modifying data from a spreadsheet file is much easier than encoding it directly at command line prompt.

Below is the text file entered at MS Excel applications. We will save JOBFILE as a CSV (comma separated values) file format.

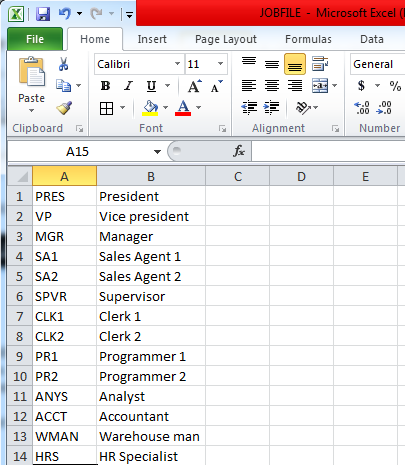


Figure 2-21. JOBFILE.CSV as an input file

Use the following syntax as guide to import text file:

IMPORT FROM <text file> OF DEL

INSERT INTO <table name>

Here you could notice that we combine IMPORT with INSERT INTO statements. OF DEL means that the file being imported uses a delimiter. On this case, the delimiter of the file is a comma.

Applying the statement, we have:

IMPORT FROM ‘C:/Users/Jerry/My Documents/JOBFILE.CSV’ OF DEL

INSERT INTO job

The text file JOBFILE.CSV includes the folder on where it is saved. Make sure you enclose it with single quotes. Below is the result of the statement after executing the IMPORT command. We list the content of job to check if insertion was done correctly.



Figure 2-22. IMPORTing and displaying the content of job table

Create the jobHistory table. Thus we define:

CREATE TABLE jobHistory (empNo VARCHAR(5) NOT NULL REFERENCES employee,

jobCode VARCHAR(4) NOT NULL REFERENCES job,

effDate DATE NOT NULL ,

salary DECIMAL(10,2) CONSTRAINT salary\_ck CHECK (salary >= 0.0),

deptCode VARCHAR(4),

PRIMARY KEY (empNo, jobCode,effDate),

FOREIGN KEY (deptCode) REFERENCES department)

empNo, jobCode, effDate are set as primary key. Columns set as primary key is called composite key. Primary key should not contain blank value that is why we indicate the NOT NULL expression. All values that entered at empNo and jobCode are cross REFERENCEd with employee and job tables. The set up increases the referential integrity of the table which means no value will be entered unless it exists on the two tables.

Salary has DECIMAL data type which has a size of 10 digits with 2 decimal point numbers. It has a CONSTRAINT with a name of salary\_ck. Only salary with values greater or equal to 0.0 are accepted.

effDate has DATE data type. This monitors the effectivity date of the position on the job history table. effDate is part of the primary key.

deptCode is set as FOREIGN KEY. We limit the entry on deptCode column which has REFERENCEd with department table. Foreign key is non-key or not primary key on one table but primary key on another table. On our case, deptCode is foreign key but referenced with department table which has its own deptCode set as primary key.

Check the structure of the table afterwards.

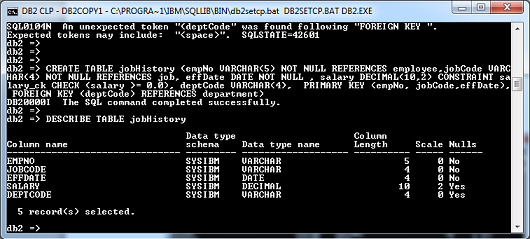
****

Figure 2-23. Structure of jobHistory table.

**Validate input row with constraints set**

Let us check if our set constraints are working. Insert the following row:

INSERT INTO employee VALUES (‘00001’, ‘Morris’, ‘Olive’, ‘F’, ‘1991-07-21’)

Notice the error message it generates:

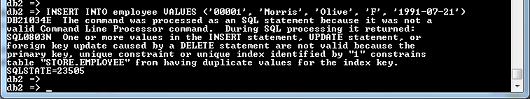


Figure 2-24. empNo with the value of 00001 is already existing

We have declared that empNo is the primary key. empNo with a value of 00001 is already existing that is why the inserted row was not accepted.

Use the arrow up key of your keyboard to re-enter the previous statement we have issued. Modify it with the following statement:

INSERT INTO employee VALUES (‘00013’, ‘Morris’, ‘Olive’, ‘f’, ‘1991-07-21’)

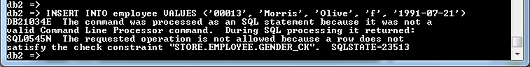


Figure 2-25. small letter f is not accepted

Though small letter f is the same with letter F, it is not accepted because big letter M and F were declared during creation of the table. Correct it with big letter F and the row will be inserted.

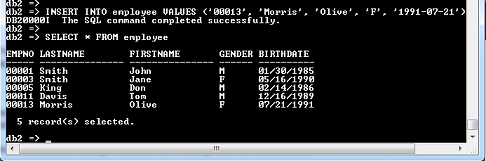


Figure 2-26. Issuing SELECT after valid INSERT statement

Work with another table. This time focus at jobHistory table. Enter the following script:

INSERT INTO jobHistory VALUES (‘00015’, ‘PR3’, ‘2010-05-11’, -50000, ‘ITC’)

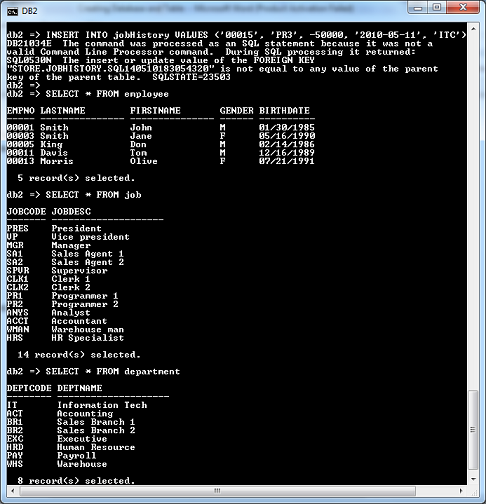


Figure 2-27. Cross-referencing with primary and foreign keys not correct

Notice on the above insert 00015, PR3, and ITC were non-existent at employee, job, and department tables. jobHistory columns are cross-REFERENCEd with these tables. Any value not included on these referenced tables will be not accepted as an entry at jobHistory.

Since the jobHistory entry is intended for John Smith who works as Programmer 2 at IT department, let us correct our insert table with the following script:

INSERT INTO jobHistory VALUES (‘00001’, ‘PR2’, ‘2010-05-11’, -50000, ‘IT’)

The following error message is generated:

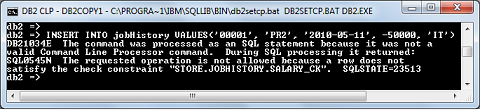


Figure 2-28. Only 0 or greater than zero is accepted for salary

Salary\_ck is the constraint name we have set before to check if value correspond with the range (negative values are not accepted). Remove the negative sign for the salary and re-enter the script.

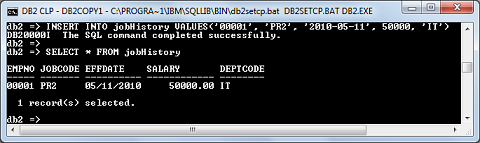


Figure 2-29. The row insertion at jobHistory is accepted

**Alter Table**

If we need to add new column we employ the ALTER command:

ALTER TABLE <table name> ADD (<column name> <data type>)

Thus, if we add hireDate and sepDate columns at employee table, we issue to separate ALTER statement:

ALTER TABLE employee ADD hireDate DATE

ALTER TABLE employee ADD sepDate DATE

Let us see the current structure of our table:

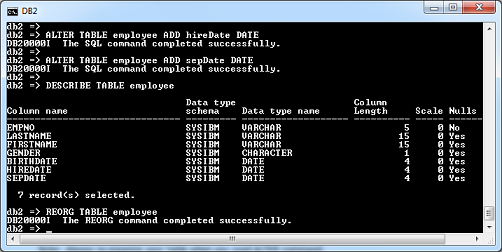


Figure 2-30. HireDate and sepDate columns added to the table

Note: Always re-organize your table when you used ALTER command:

REORG TABLE <table name>

On this case, we issue:

REORG TABLE employee

**Update Table**

You can update the content of each column with the following UPDATE syntax:

UPDATE <table name> SET <column name> = <new value>

<filter condition>

SET specifies which column to update and assign it with a new value after the equal sign.

Use WHERE clause as filter condition when updating rows. On our example, empNo column with its corresponding data value will be used to select which row to update.

We will now update our rows hireDate with the UPDATE command:

UPDATE employee SET hireDate = ‘2010-05-11’

WHERE empNo = ’00001’

UPDATE employee SET hireDate = ‘2010-05-11’

WHERE empNo = ’00003’

UPDATE employee SET hireDate = ‘2010-06-23’

WHERE empNo = ’00005’

UPDATE employee SET hireDate = ‘2010-06-30’

WHERE empNo = ’00011’

UPDATE employee SET hireDate = ‘2010-06-30’

WHERE empNo = ’00013’

Employee Don King did not stay with the company. Update his separation date with the following statement:

UPDATE employee SET sepDate = ‘2014-01-15’

WHERE empNo = ‘00005’

Display all rows after the UPDATEs:

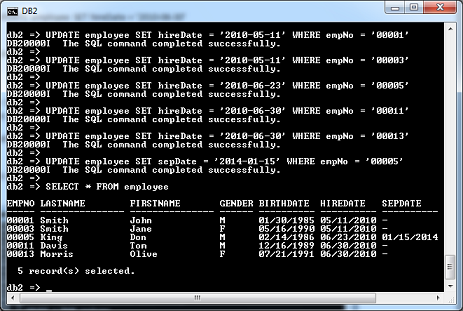


Figure 2-31. Using UPDATE statement

Failute to use WHERE condition will update column hiredate on all rows with the value set.

**Delete rows of the table**

Same with UPDATE statement, you need to set a condition on what rows should be deleted:

DELETE FROM <table name>

<condition>

Considering we delete all rows with a hireDate of 06/30/2010, we issue:

DELETE FROM employee

WHERE hireDate =’2010-06-30’

Let us check our table:



Figure 2-32. DELETE statement

Same with UPDATE statement, if you fail to include WHERE clause the entire rows of the table will be deleted.

**DROP database, table, and column**

DROP statement is executed to wipe out either a database, table, or column. Use it with caution because once implemented it is no longer reversible. Below is the syntax on how to use DROP:

DROP DATABASE <database name>

DROP TABLE <table name>

ALTER TABLE <table name> DROP COLUMN <column name>

**Chapter Exercise**

I**. Multiple choice**: Select the letter that corresponds to the correct answer.

1. What key that specifies the uniqueness of a row instance?
2. FOREIGN KEY
3. PRIMARY KEY
4. TRANSITIVE KEY
5. COMPOSITE KEY
6. How do you open an existing database SAMPLE?
7. CONNECT DATABASE sample
8. CONNECT sample
9. CONNECT TO DATABASE sample
10. CONNECT TO sample
11. What do you call more than one column sets as a primary key of a specific table?
12. FOREIGN KEY
13. Transitive dependency
14. Functional dependency
15. Composite key
16. What do you do after you ALTER TABLE by dropping or adding column to its structure?
17. REORG TABLE
18. RE INDEX TABLE
19. DESCRIBE TABLE
20. None of the above
21. SEQUENCE transNo\_seq was CACHE by 3 and START with the value 2. The sequence INCREMENTs only by 1. Considering the connection to the database was interrupted because of transmission problem, what would be the NEXT VALUE of the sequence when the last value was set to 3?
22. 4
23. 5
24. 6
25. 3
26. How do you DELETE row with lastName equal to ‘Thompson’ from STUDENT table?
27. DELETE student WHERE lastName IS ‘Thompson’
28. DELETE student WHERE lastName = ‘Thompson
29. DROP FROM student WHERE lastName = ‘Thompson’
30. DELETE FROM student WHERE lastName = ‘Thompson’
31. What command do you execute to see the column data types of table PRODUCT?
32. SELECT \* FROM product
33. SELECT DATA TYPES FROM product
34. DESCRIBE TABLE product
35. CONNECT TABLE product
36. How do you ensure that only category freshmen, sophomore, junior, and senior are accepted for the column yearLevel? Assume that the first three letter in upper case letters of the word serves as the code
37. yearLevel CHAR(3) CONSTRAINT year\_ck CHECK (yearLevel IN (‘FRE’, ’SOP’, ’JUN’, ‘SEN’))
38. yearLevel CHAR(3) CONSTRAINT CHECK (yearLevel IN (‘FRE’, ’SOP’, ’JUN’, ‘SEN’))
39. yearLevel CHAR(3) CONSTRAINT year\_ck CHECK (yearLevel IN ‘fre’, ‘sop’, ‘jun’)
40. yearLevel CHAR(3)
41. The company imposes a 10% increase of salary to all employees. What command you need to execute?
42. UPDATE employee SET salary = salary \* 0.10
43. UPDATE employee SET salary = salary + salary \* 0.10
44. UPDATE employee SET salary = salary \* 0.10 WHERE empNo EQUAL TO ALL
45. UPDATE employee SET salary = salary + salary \* 0.10 WHERE empNo NOT NULL
46. prodCode with a size of 6 is set as primary key of table PRODUCT. How do you specify this part in CREATE TABLE statement?
47. prodCode VARCHAR(6) PRIMARY KEY
48. prodCode VARCHAR(6) NOT NULL PRIMARY KEY
49. prodCode VARCHAR(6) COMPOSITE KEY
50. prodCode VARCHAR(6) NOT NULL FOREIGN KEY

II. **Problems**

1. Insert EMPFILE.CSV and JOBHISTFILE.CSV to employee, jobHistory, respectively. You may download EMPFILE.CSV to this link; JOBHISTFILE.CSV to this link. If you choose to type it on spreadsheet by yourself, refer to the following:



Figure 2-33. EMPFILE.CSV

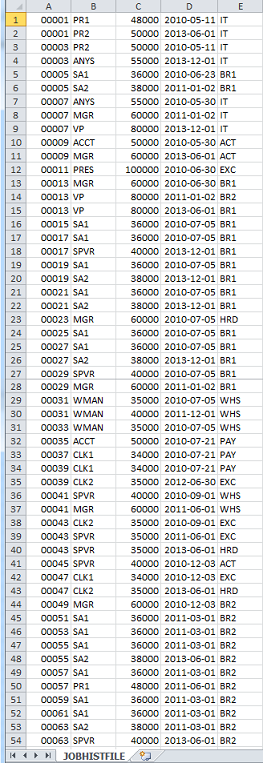


Figure 2-34. JOBHISTFILE.CSV

NOTE: If you download the files, refrain from opening and saving it using MS Excel to avoid change of formats.

1. The ERD of Hope, Inc. is updated with another sets of tables that records the transactions of sales agents file. Create the new sets of table with its corresponding formats. Do parent tables first as you proceed on child tables next.

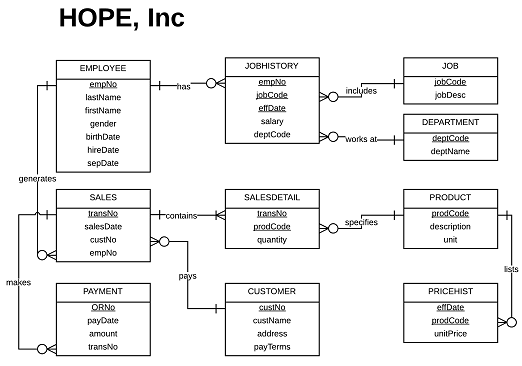


Figure 2-35. Updated ERD of Hope, Inc.

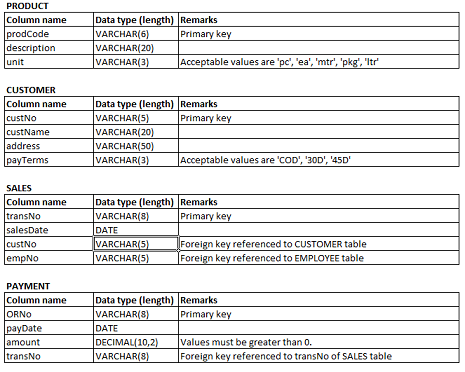


Figure 2-36. Parent product, customer, sales and payment tables

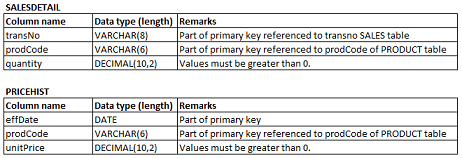


Figure 2-37. Child salesDetail and priceHist tables

1. Insert PRODFILE.CSV, CUSTFILE.CSV, SALES.CV, SALESDFILE.CSV, and PRHIST. CSV files to its respective tables. Click to the following links to download:.
2. PRODFILE.CSV
3. CUSTFILE.CSV
4. SALESFILE.CV
5. SALESDFILE.CSV
6. PRICEHISTFILE.CSV

NOTE: If you download the files, refrain from opening and saving it using MS Excel to avoid change of formats.

**Answer to selected numbers**

I. Multiple choice

1. B

3. D

5. B

7. C

9. B