Practice 2: DDL

Jesús Manuel Juárez Pasillas

August 27, 2021

1 Introduction

Knowing basic concepts of theory and practice on databases allows us to apply these concepts correctly and consequently make an optimal database without making basic mistakes. The ddl statements allow us to make transactions for a desired schema, such as creating or deleting a table, inserting, modifying or deleting data on a table, etc.

2 Developing

2.1 Activity 1:

This activity consists of choosing the correct answer or answers, in addition to arguing the answer.

1. If a table is created without specifying a schema, in which schema will it be? (Choose the best answer). Challenge question.

   A. It will be an orphaned table, without a schema.

   B. The creation will fail.

   C. It will be in the SYS schema.

   D. It will be in the schema of the user creating it.

   E. It will be in the PUBLIC schema.

   - Since it is not indicated which user is the one who wants to create the table, slq takes the current user.

2. Several object types share the same namespace, and therefore cannot have the same name in the same schema. Which of the following object types is not in the same namespace as the others? (Choose the best answer). Challenge question.

   A. Index

   B. PL/SQL stored procedure

C. Synonym

D. Table

E. View

- The index belongs to a namespace.

3. Which of these statements will fail because the table name is not legal? (Choose two answers.)

A. create table “WHERE” (col1 date);

B. create table “mincase” (col1 date);

C. create table 1var (col1 date);

D. create table var1 (col1 date);

E. create table delete (col1 date);

- You cannot put a number at the beginning of a table name, and you cannot have reserved names.

4. 4. Which of the following data types are variable length? (Choose all correct answers.)

A. BLOB

B. CHAR

C. LONG

D. NUMBER

E. RAW

F. VARCHAR2

- The selected data types are not given their length so they can be of different lengths.

5. Study these statements:
create table tab1 (c1 number(1), c2 date);
alter session set nls_date_format='dd-mm-yy';
insert into tab1 values (2.2,'29-07-09');
Will the insert succeed? (Choose the best answer)

(a) The insert will fail because the 2.2 is too long.

(b) The insert will fail because the '29-07-09' is a string, not a date.

(c) The insert will fail for both reasons A and B.

(d) The insert will succeed.

- The insertion is done correctly because the format in which the insertion is made is correct, a number and the date in "dd-mm-yy" format.

6. Which of the following is not supported by Oracle as an internal data type? (Choose the best answer.)

   A. CHAR

   B. FLOAT

   C. INTEGER

   D. STRING

   - The String data type does not exist in Oracle.

7. Consider this statement:
   create table t1 as select * from employees where 9=4;
   What will be the result? (Choose the best answer.)

   A. There will be an error because of the impossible condition.

   B. No table will be created because the condition returns FALSE.

   C. The table T1 will be created but no rows inserted because the conditionreturns FALSE.

   D. The table T1 will be created and every row in EMPLOYEES inserted because the condition returns a NULL as a row filter.

   - The table will be created, but without any data, regardless of whether the where condition returns false.

8. When a table is created with a statement such as the following:
   create table newtable as select * from oldtable; will there be any constraints on the new table? (Choose the best answer.)

   A. The new table will have no constraints, because constraints are not copied when creating tables with a subquery.

   B. All the constraints on TAB will be copied to NEWTABLE.

   C. Primary key and unique constraints will be copied but not check and not null constraints.

   D. Check and not null constraints will be copied but not unique or primary key.

   E. All constraints will be copied, except foreign key constraints.

   - If a table is made based on another, in the new table it will not have the restrictions of the old table, it will only have its fields.

9. Which types of constraint require an index? (Choose all that apply.)

   A. CHECK

   B. NOT NULL

   C. PRIMARY KEY

D. UNIQUE

- UNIQUE and PRIMARY KEY constraints generate indexes that enforce or "back" the constraint.

10. A transaction consists of two statements. The first succeeds, but the second (which updates several rows) fails partway through because of a constraint violation. What will happen? (Choose the best answer). Challenge question.

    A. The whole transaction will be rolled back.

    B. The second statement will be rolled back completely, and the first will be committed.

    C. The second statement will be rolled back completely, and the first will remain uncommitted.

    D. Only the one update that caused the violation will be rolled back; everything else will be committed.

    E. Only the one update that caused the violation will be rolled back; everything else will remain uncommitted.

    - If there is an error in the transaction, no matter how many statements it contains, the transaction will fail and do nothing.

2.2 Activity 2:

For this activity, you are asked to follow the steps in each section of the activity and take screenshots of the results.

2.2.1 Determine What Objects Are Accessible to Your Session.

In this exercise, query various data dictionary views as user HR to determine what objects are in the HR schema and what objects in other HR schemas has access to.

1. Connect to the database with SQL Developer as user HR.

2. Determine how many objects of each type are in the HR schema (Figure 1):
   select object_type,count(*) from user_objects group by object_type;
   The USER_OBJECTS view lists all objects owned by the schema to which the current session is connected, in this case HR.

	OBJECT_TYPE	COUNT(*)
1	SEQUENCE	3
2	PROCEDURE	2
3	TRIGGER	2
4	TABLE	7
5	INDEX	19
6	VIEW	1

Figure 1: How many objects in the HR schema.

3. Determine how many objects in total HR has permissions on:
   select object_type,count(*) from all_objects group by object_type;
   The ALL_OBJECTS view lists all objects to which the user has some sort of access.

	OBJECT_TYPE	COUNT(*)
1	EDITION	1
2	CONSUMER GROUP	2
3	SEQUENCE	12
4	SCHEDULE	3
5	PROCEDURE	31
6	OPERATOR	45
7	DESTINATION	2
8	WINDOW	9
9	SCHEDULER GROUP	4
10	PACKAGE	296
11	PROGRAM	11
12	XML SCHEMA	31
13	JOB CLASS	2
14	TRIGGER	2
15	TABLE	94
16	SYNONYM	3975
17	VIEW	1579
18	FUNCTION	163
19	INDEXTYPE	8
20	INDEX	19
21	TYPE	995
22	EVALUATION CONTEXT	1

Figure 2: Objects with permissions on.

4. Determine who owns the objects HR can see:
   select distinct owner from all_objects;

	OWNER
1	MDSYS
2	PUBLIC
3	CTXSYS
4	HR
5	SYSTEM
6	APEX_040000
7	XDB
8	SYS

Figure 3: Owner of the objects that HR.

2.2.2 Investigate Table Structures.

In this exercise, query various data dictionary views as user HR to determine the structure of a table.

1. Connect to the database with SQL Developer as user HR.
2. Determine the names and types of tables that exist in the HR schema:
   select table_name,cluster_name,iot_type from user_tables;
   Clustered tables and index organized tables (IOTs) are advanced table structures. In the HR schema, all tables are standard heap tables except for COUNTRIES which is an IOT.

	TABLE_NAME	CLUSTER_NAME	IOT_TYPE
1	COUNTRIES	(null)	IOT
2	JOB_HISTORY	(null)	(null)
3	EMPLOYEES	(null)	(null)
4	JOBS	(null)	(null)
5	DEPARTMENTS	(null)	(null)
6	LOCATIONS	(null)	(null)
7	REGIONS	(null)	(null)

Figure 4: Table names and types in HR schema.

3. Use the DESCRIBE command to display the structure of a table:
   describe regions;

```
Nombre      ¿Nulo?   Tipo
----------- -------- ------------
REGION_ID   NOT NULL NUMBER
REGION_NAME          VARCHAR2(25)
```

Figure 5: describe regions.

4. Retrieve similar information by querying a data dictionary view:
   select column_name,data_type,nullable from user_tab_columns where table_name='REGIONS';

	COLUMN_NAME	DATA_TYPE	NULLABLE
1	REGION_ID	NUMBER	N
2	REGION_NAME	VARCHAR2	Y

Figure 6: Query a data dictionary view.

2.2.3 Investigate the Data Types in the HR schema.

In this exercise, find out what data types are used in the tables in the HR schema, using two techniques.

1. Connect to the database as user HR with SQL Developer.
2. Use the DESCRIBE command to show the data types in some tables:
   describe employees;
   describe departments;

```
Nombre         ¿Nulo?   Tipo
-------------- -------- ------------
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)
```

```
Nombre          ¿Nulo?   Tipo
--------------- -------- ------------
DEPARTMENT_ID   NOT NULL NUMBER(4)
DEPARTMENT_NAME NOT NULL VARCHAR2(30)
MANAGER_ID               NUMBER(6)
LOCATION_ID              NUMBER(4)
```

Figure 7: Query a data dictionary view.

3. Use a query against a data dictionary view to show what columns make up the EMPLOYEES table, as the DESCRIBE command would:
   select column_name,data_type,nullable,data_length,data_precision, data_scale from user_tab_columns where table_name='EMPLOYEES';

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_LENGTH	DATA_PRECISION	DATA_SCALE
1	EMPLOYEE_ID	NUMBER	N	22	6	0
2	FIRST_NAME	VARCHAR2	Y	20	(null)	(null)
3	LAST_NAME	VARCHAR2	N	25	(null)	(null)
4	EMAIL	VARCHAR2	N	25	(null)	(null)
5	PHONE_NUMBER	VARCHAR2	Y	20	(null)	(null)
6	HIRE_DATE	DATE	N	7	(null)	(null)
7	JOB_ID	VARCHAR2	N	10	(null)	(null)
8	SALARY	NUMBER	Y	22	8	2
9	COMMISSION_PCT	NUMBER	Y	22	2	2
10	MANAGER_ID	NUMBER	Y	22	6	0
11	DEPARTMENT_ID	NUMBER	Y	22	4	0

Figure 8: Consult data dictionary.

2.2.4 Create Tables.

In this exercise, use SQL Developer to create a heap table, insert some rows with a subquery, and modify the table. Do some more modifications with SQL*Plus, then drop the table (A new image of the steps is attached).

1. Connect to the database as user HR with SQL Developer.
2. Right-click the Tables branch of the navigation tree, and click New Table.
3. Name the new table EMPS, and use the Add Column button to set it up as in the following illustration:

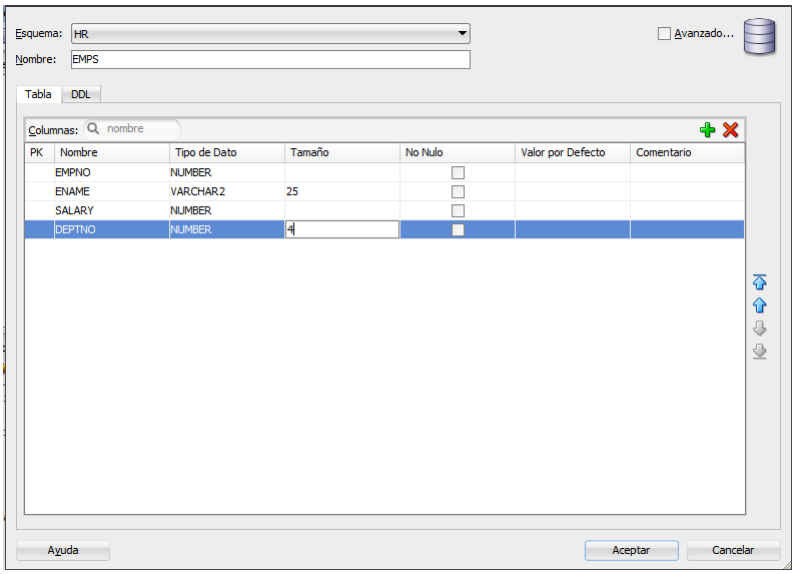


Figure 9: Create table.

4. Click the DDL tab to see if the statement that has been constructed. It should look like this:
   CREATE TABLE EMPS(
   EMPNO NUMBER,
   ENAME VARCHAR2(25),
   SALARY NUMBER,
   DEPTNO NUMBER(4, 0));
   Return to the Table tab (as in the preceding illustration) and click OK to create the table.

5. Run this statement:
   insert into emps select employee_id,last_name,salary,department_id from employees;
   and commit the insert:
   commit;

6. Right-click the EMPS table in the SQL Developer navigator, click Column and Add.

7. Define a new column HIRED, type DATE, as in the following illustration below; and click Apply to create the column.

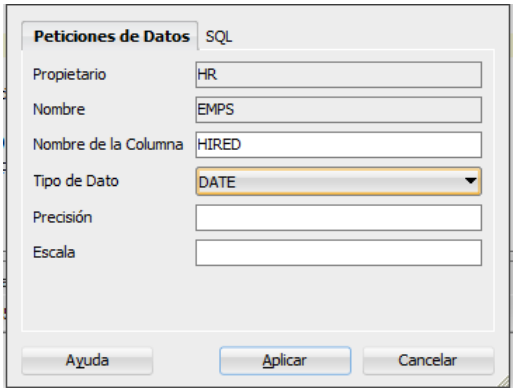


Figure 10: New column.

8. Connect to the database as HR.

9. Define a default for HIRED column in the EMPS table:
   alter table emps modify (hired default sysdate);

10. Insert a row without specifying a value for HIRED and check that the new row does have a HIRED date but that the other rows do not:
    insert into emps (empno,ename) values(99,'Newman');
    select hired,count(1) from emps group by hired;

	HIRED	COUNT(1)
1	(null)	107
2	24/08/21	1

Figure 11: Insert into emps.

11. Tidy up by dropping the new table:
drop table emps;

2.2.5 Work with Constraints.

Use SQL Developer to create tables, add constraints, and demonstrate their use.

1. Connect to the database as user HR.
2. Create a table EMP as a copy of some columns from EMPLOYEES:
create table emp as select employee_id empno, last_name ename, department_id deptno from employees;
3. Create a table DEPT as a copy of some columns from DEPARTMENTS:
create table dept as select department_id deptno, department_name dname from departments;
4. Use DESCRIBE to describe the structure of the new tables. Note that the not null constraint on ENAME and DNAME has been carried over from the source tables.

```
Nombre ¿Nulo?   Tipo
------ -------- ------------
EMPNO           NUMBER(6)
ENAME  NOT NULL VARCHAR2(25)
DEPTNO          NUMBER(4)

Nombre ¿Nulo?   Tipo
------ -------- ------------
DEPTNO          NUMBER(4)
DNAME  NOT NULL VARCHAR2(30)
```

Figure 12: Describe emp and dept.

5. Add a primary key constraint to EMP and to DEPT and a foreign key constraint linking the tables:
alter table emp add constraint emp_pk primary key (empno);
alter table dept add constraint dept_pk primary key (deptno);
alter table emp add constrain dept_fk foreign key (deptno) references dept on delete set null;
The preceding last constraint does not specify which column of DEPT to reference; this will default to the primary key column.
6. Demonstrate the effectiveness of the constraints by trying to insert data that will violate them:
insert into dept values(10,'New Department');
insert into emp values(9999,'New emp',99); truncate table dept;

```
Error que empieza en la línea: 17 del comando :
insert into dept values(10,'New Department')
Informe de error -
ORA-00001: unique constraint (HR.DEPT_PK) violated


Error que empieza en la línea: 18 del comando :
insert into emp values(9999,'New emp',99)
Informe de error -
ORA-02291: integrity constraint (HR.DEPT_FK) violated - parent key not found
```

Figure 13: Inserts.

```
Error que empieza en la línea: 19 del comando :
truncate table dept
Informe de error -
ORA-02266: unique/primary keys in table referenced by enabled foreign keys
02266. 00000 -  "unique/primary keys in table referenced by enabled foreign keys"
*Cause:    An attempt was made to truncate a table with unique or
           primary keys referenced by foreign keys enabled in another table.
           Other operations not allowed are dropping/truncating a partition of a
           partitioned table or an ALTER TABLE EXCHANGE PARTITION.
*Action:   Before performing the above operations the table, disable the
           foreign key constraints in other tables. You can see what
           constraints are referencing a table by issuing the following
           command:
           SELECT * FROM USER_CONSTRAINTS WHERE TABLE_NAME = "tabnam";
```

Figure 14: Truncate.

7. Tidy up by dropping the tables. Note that this must be done in the correct order:
   drop table emp;
   drop table dept;

2.3 Activity 3:

Consider this simple analysis of a call record system in a local telephone company:
A subscriber is identified by a customer number and also has a name, last name, date of birth, address, rfc, 1 or 2 references (friends or relatives) and possibly one or more telephones. A telephone is identified by its number, which must be a 10-digit integer beginning with 55, an activation date, and a flag ('A' or 'I') for whether it is active. Inactive telephones are not assigned to a subscriber; active telephones are. These subscribers are associated to physical telephones, and also have a brand, capacity (memory, cpu, display, camera, ...) and serial number. Besides, all telephones are engaged with a forced plan: 6, 12, 18 or 24 months. It is necessary to store the start and final date of it (when this is hired). For every call, it is necessary to record the time it started and the time it finished.

- Create the necessary tables.
  - create table customers(
    customer_id number(8) constraint customer_pk primary key,
    name varchar(25) constraint cus_name_nn not null,
    last_name varchar(25) constraint cus_last_name_nn not null,
    date_birth date,
    address varchar(100) constraint cus_address_nn not null,
    rfc char(13) constraint cus_rfc_nn not null,
    reference1 varchar(50) constraint cus_ref1_nn not null,
    reference2 varchar(50));
  - create table telephone_numbers(
    telephone_number number(10) constraint telephone_pk primary key,
    activation date constraint tel_activation_nn not null,
    flag char(1) constraint tel_flag_nn not null,
    constraint tel_flag_check check (flag in('A','I')));
  - create table customers_tel_numbers(
    customer_id number(8),
    telephone_number number(10),
    constraint customer_tel_number_pk primary key (customer_id, telephone_number),
    constraint cust_cust_tel_number_fk foreign key (customer_id) references customers(customer_id),
    constraint tel_cust_tel_number_fk foreign key (telephone_number) references telephone_numbers(telephone_number));
  - create table telephones(
    serial_number number(16) constraint py_telephone_pk primary key,
    brand varchar(20) constraint tel_brand_nn not null,
    tel_capacity number(5,2) constraint tel_tel_capacity_nn not null,
    cpu varchar(50) constraint tel_cpu_nn not null,
    display varchar(50) constraint tel_display_nn not null,
    camera number(2) constraint tel_camera_nn not null);
  - create table customer_telephones(
    customer_id number(8),
    serial_number number(16),
    constraint customer_telephone_pk primary key (customer_id,serial_number),
    constraint cust_cust_tel_fk foreign key (customer_id) references customers(customer_id),
    constraint tel_cus_tel_fk foreign key (serial_number) references telephones(serial_number));
  - create table types_plan(
    plan_id number(3) constraint type_plan_pk primary key,
    duration char(2) constraint type_plan_duration_nn not null,
    constraint type_plan_duration_check check(duration in(6,12,18,24)));

- create table telephone_plan(
  telephone_number number(10),
  plan_id number(3),
  constraint telephone_plan_pk primary key (telephone_number,plan_id),
  constraint tel_plan_tel_number_fk foreign key (telephone_number) references telephone_numbers(telephone_number),
  constraint type_plan_telephone_plan_fk foreign key (plan_id) references types_plan(plan_id));
- create table calls(
  telephone_number1 number(10),
  telephone_number2 number(10),
  time_started date,
  time_end date,
  constraint call_pk primary key (telephone_number1,telephone_number2, time_started,time_end),
  constraint tel1_calls_fk foreign key (telephone_number1) references telephone_numbers(telephone_number),
  constraint tel2_calls_fk foreign key (telephone_number2) references telephone_numbers(telephone_number));

- Generate the constraints and defaults that can be used to implement this system.
- Generate the corresponding relational model using SQL Data Modeler (dragging the tables using GUI).

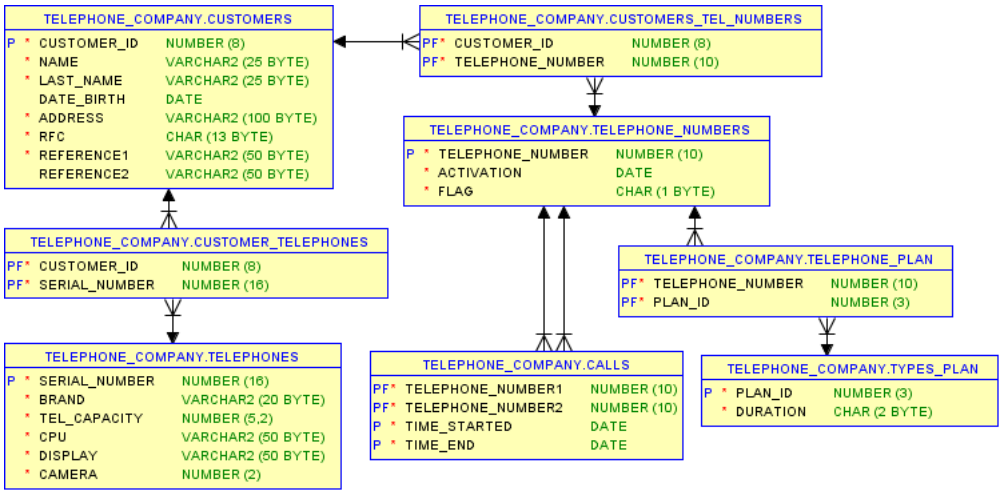


Figure 15: Relational model.

- Generate the corresponding logical model using SQL Data Modeler (reverse engineering).

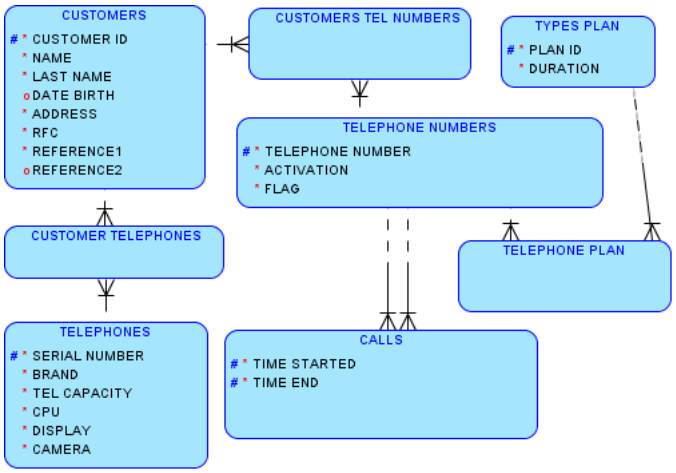


Figure 16: Logical model.

2.4 Activity 4:

Propose a response to the following scenario issue:

- You are designing table structures for a human resources application. The business analysts have said that when an employee leaves the company, his employee record should be moved to an archive table. Can constraints help? Explain the reasons.

Constraints help maintain the integrity of the database, so they would help make the transaction very easy.

2.5 Activity 5:

Carry out the following steps (capture an image for each statement output):

- Create the EMPLOYEES_2 table based on the EMPLOYEES table from HR scheme. Use the CREATE statement that employs a SELECT statement.
  - create table employees_2 as (select * from employees);

- Describe the table structure.

```
Nombre         ¿Nulo?   Tipo
-------------- -------- ------------
EMPLOYEE_ID             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)
```

Figure 17: Describe table.

- Alter the EMPLOYEES_2 table status to read-only.
  - alter table employees_2 read only;
- Try to insert a row the table. Depict the results.

```
Error que empieza en la linea: 7 del comando -
insert into employees_2 values(43,'Juan','Perez','juan@gmailcom',null,'27-08-21','AD_PRES',null,null,null,null)
Error en la linea de comandos : 7 Columna : 13
Informe de error -
Error SQL: ORA-12081: update operation not allowed on table "HR"."EMPLOYEES_2"
12081. 00000 -  "update operation not allowed on table \"%s\".\"%s\""
*Cause:    An attempt was made to update a read-only materialized view.
*Action:   No action required. Only Oracle is allowed to update a
           read-only materialized view.
```

Figure 18: Insert failed.

- Revert the EMPLOYEES_2 table to the write status. Now, try to insert the same row again. Depict the results.
  - alter table employees_2 read write;

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
43	Juan	Perez	juan@gmailcom	(null)	27/08/21	AD_PRES	(null)	(null)	(null)	(null)

Figure 19: Insert.

  -

- Drop the EMPLOYEES_2 table.
  - drop table employees_2;

2.6 Activity 6:

Taking into account the following diagram.

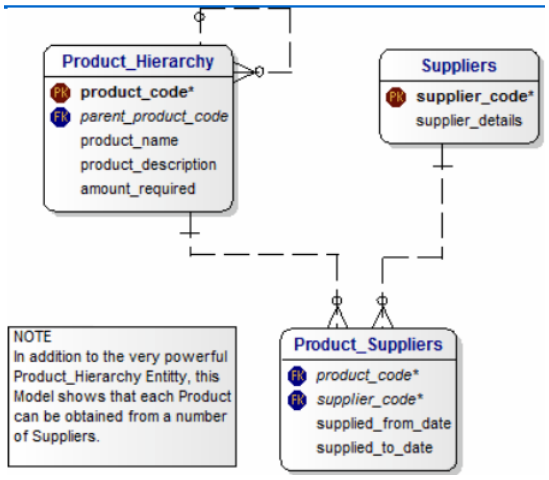


Figure 20: Diagram.

1. Generate the DDL statements to create only the isolated tables of Product_Hierarchy and Product_Suppliers (add constraints here in CREATE, don‘t add primary and foreign keys yet). Use descriptive names in constraints.

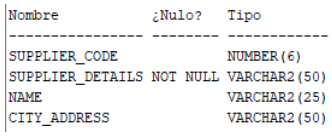
   - create table Suppliers(
     supplier_code number(6),
     supplier_details varchar(50) constraint suppliers_supp_details_nn not null);
   - create table Product_Supplier(
     product_code number(6),
     supplier_code number(6),
     supplied_from_date date constraint product_supplier_from_nn not null,
     supplied_to_date date constraint product_supplier_to_nn not null);
   - create table Product_Hierarchy(
     product_code number(6),
     parent_product_code number(6),
     product_name varchar(25),
     product_description varchar(75),
     amount_required number(3));

2. Generate the DDL statements to create the Suppliers table.

3. Add the fields: “status” as numeric, ”name” as varchar and “city” as int.

- alter table Suppliers add(status numeric(1),name varchar(25),city int);

4. Delete the column “status” from the Suppliers table.
   - alter table Suppliers drop column status;
5. Rename the column ”city” to be called ”city_address”.
   - alter table Suppliers rename column city to city_address;
6. Modify the data type of the “city_address” column, so that instead of saving numbers, it saves a variable string with a maximum length of 50.
   - alter table Suppliers modify city_address varchar(50);
7. Shows the structure of the Suppliers table.



Nombre	¿Nulo?	Tipo
SUPPLIER_CODE		NUMBER(6)
SUPPLIER_DETAILS	NOT NULL	VARCHAR2(50)
NAME		VARCHAR2(25)
CITY_ADDRESS		VARCHAR2(50)

Figure 21: Describe suppliers.

8. Generate, with the ALTER statements, all the necessary instructions to link all the tables as they appear in the diagram.
   - alter table Suppliers add constraint suppliers_pk primary key (supplier_code);
   - alter table Product_Hierarchy add constraint hroduct_hierarchy_pk primary key (product_code);
   - alter table Product_Hierarchy add constraint Product_Hierarchy_fk foreign key (parent_product_code) references Product_Hierarchy(product_code);
   - alter table Product_Supplier add constraint Product_Product_suppliers_fk foreign key (product_code) references Product_Hierarchy(product_code);
   - alter table Product_Supplier add constraint Suppliers_Product_suppliers_fk foreign key (supplier_code) references Suppliers(supplier_code);

9. Obtain the relational diagram and compare the results. If necessary, modify your script to suit the correct one. The diagrams remained the same,

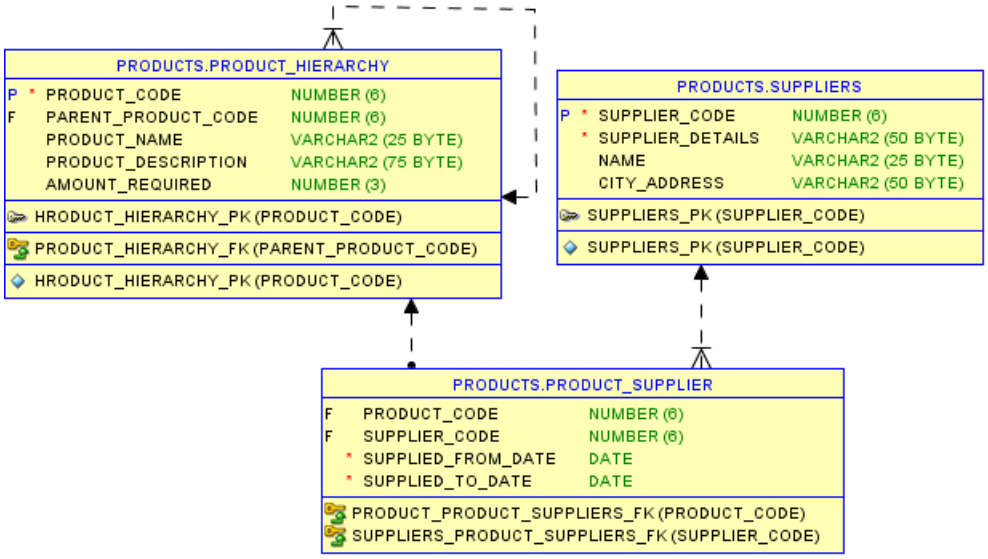


Figure 22: Relational model.

relations, primary keys and foreign keys. When creating the primary and foreign key constraints, having created the tables only with the not null constraints, it was concluded that this is the best way to create the tables in a schema.

2.7 Activity 7:

Complete the following ER diagram with the corresponding attributes. The main idea is: an employee (with their personal data) works on a specific job, which initiated in a particular date/hour with a description and status. This job is carried out in a branch, in which the task is located. Thus, a supervisor manages the entire task with a specific date of assignment.

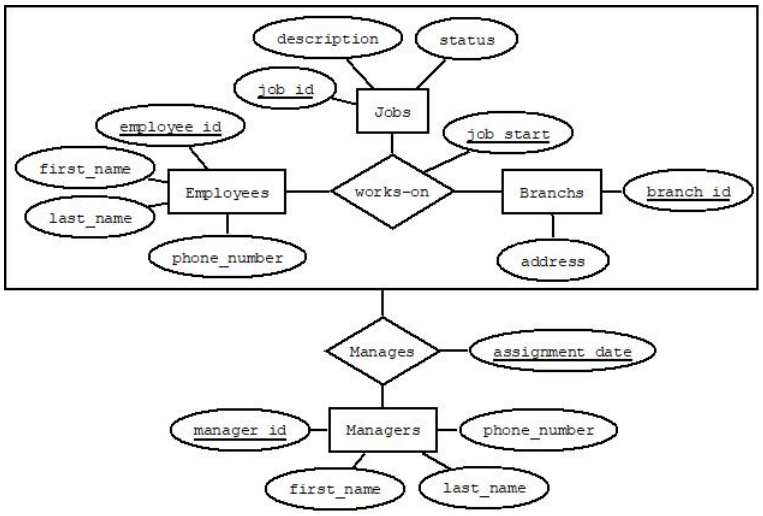


Figure 23: Diagram E-R.

Generate the corresponding DDL statements. Don't forget to consider:

- CREATE table statements are the first to be applied. In this case, the basic constraints must be performed here: not null, default, ...
  - create table jobs(
    job_id number(6),
    job_description varchar(50) constraint jobs_job_description_nn not null,
    status numeric(1) constraint jobs_status_nn not null);
  - create table employees(
    employee_id number(6),
    frist_name varchar2(25) constraint employees_name_nn not null,
    last_name varchar2(25) constraint employees_last_name_nn not null,
    phone_number numeric(10) constraint employees_phone_nn not null);
  - create table branchs(
    branch_id number(6),
    address varchar2(75) constraint branchs_address_nn not null);
  - create table work_on(
    employee_id number(6),
    job_id number(6),
    branch_id number(6),
    job_start date);
  - create table managers(
    manager_id number(6),
    frist_name varchar2(25) constraint managers_name_nn not null,
    last_name varchar2(25) constraint managers_last_name_nn not null,
    phone_number numeric(10) constraint managers_phone_nn not null);

- create table manages(
  employee_id number(6),
  job_id number(6),
  branch_id number(6),
  job_start date,
  assignment_date date,
  manager_id number(6));

- Use ALTER table statements to add primary and foreign keys.
  - alter table jobs add constraint job_pk primary key (job_id);
  - alter table employees add constraint employee_pk primary key (employee_id);
  - alter table branchs add constraint branch_pk primary key (branch_id);
  - alter table work_on add constraint work_on_pk primary key (employee_id,job_id,branch_id,job_start);
  - alter table work_on add constraint employee_work_on_fk foreign key (employee_id) references employees(employee_id);
  - alter table work_on add constraint job_work_on_fk foreign key (job_id) references jobs(job_id);
  - alter table work_on add constraint branch_work_on_fk foreign key (branch_id) references branchs(branch_id);
  - alter table managers add constraint manager_pk primary key (manager_id);
  - alter table manages add constraint manages_pk primary key (employee_id,job_id,branch_id,job_start,assignment_date,manager_id);
  - alter table manages add constraint manager_manages_fk foreign key (manager_id) references managers(manager_id);
  - alter table manages add constraint work_on_manages_fk foreign key (employee_id, job_id, branch_id, job_start) references work_on(employee_id, job_id, branch_id, job_start);

- Generate automatically the corresponding relational model using Data Modeler.

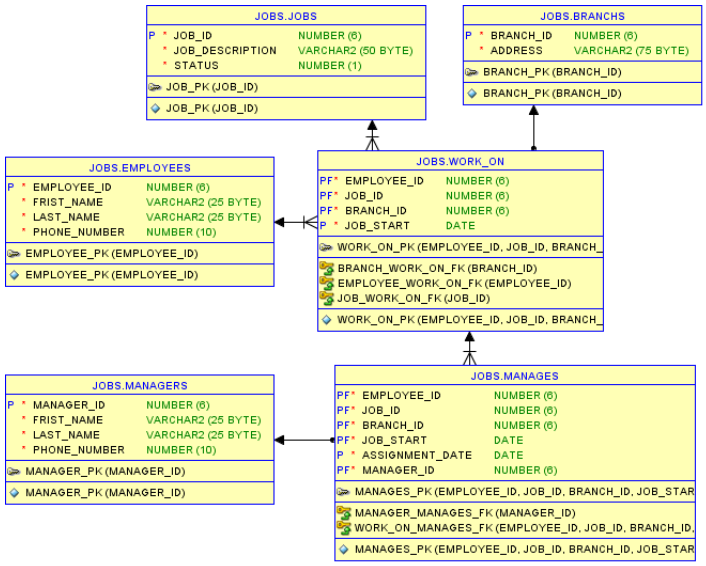


Figure 24: Relational model.

- Generate the logical diagram in SQL Data Modeler.

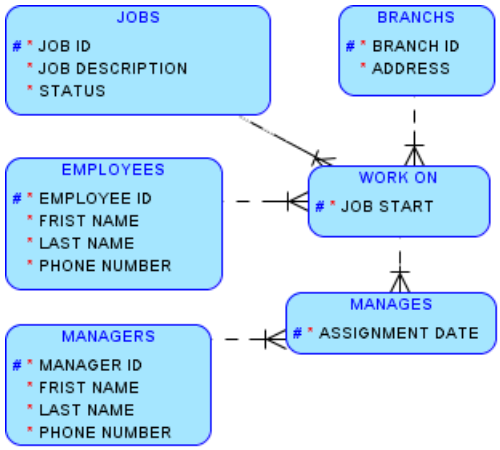


Figure 25: Logical model.

- Compare the results of the ER diagram and the logical diagram.
  The E-R diagram and the logic model are very similar in terms of what

they present, since they only show the data they have on their own and not the foreign data, and the relationships are totally the same.

2.8 Pre-Assessment:

- Practices pre-Assessment for Database Systems Laboratory II

Practice	Pre-Assessment
COMPLIES WITH THE REQUESTED FUNCTIONALITY	
HAS THE CORRECT INDENTATION	X
HAS AN EASY WAY TO ACCESS THE PROVIDED FILES	X
HAS A REPORT WITH IDC FORMAT	X
REPORT INFORMATION IS FREE OF SPELLING ERRORS	X
DELIVERED IN TIME AND FORM	X
IS FULLY COMPLETED (SPECIFY THE PERCENTAGE COMPLETED)	90

Table 1: Pre-Assessment.

2.9 Conclusion:

The basic concepts for creating and modifying a set of tables in a database, as well as for their subsequent insertion, modification or deletion of data allows us to do these transactions in a much less complicated way. For this, it is necessary to know well the creation of tables with the "create table" statement and the possibilities that it has, as well as the "alter table" statement to modify columns (this includes adding or removing columns, adding or removing restrictions, etc.) and many more sentences.