ESCUELA
COLOMBIANA
DE INGENIERÍA
JULIO GARAVITO

GUIA DE AUTOESTUDIO 3

SQL Developer

ALUMNOS:

Ignacio Andrés Castillo Rendon

Anderson Fabian Garcia Nieto

Laboratorio-Modelos de bases de datos
2024-2

DOCENTE:
MARIA IRMA DIAZ ROZO

Bogotá D.C-Colombia

1. APRENDIENDO A CREAR, ACTUALIZAR Y BORRAR TABLAS
A ADICIONAR, ACTUALIZAR Y ELIMINAR TUPLAS
A CREAR Y BORRAR VISTAS
A CREAR Y BORRAR ÍNDICES
CONSULTAR LOS DATOS SOBRE LOS DATOS.

SQL CREATE TABLE

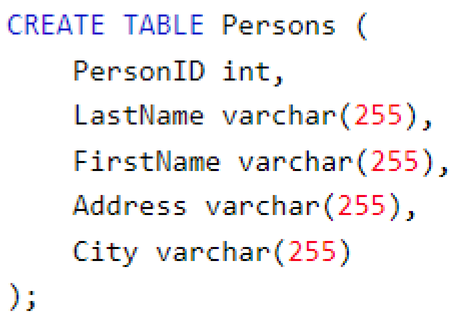
- **Crear una tabla normal**

Para crear una tabla en SQL, necesitaremos de la siguiente SINTAXIS:

```
CREATE TABLE (nombre de la tabla) (
                    name_col1 tipo de dato,
                    name_col2 tipo de dato,
                    name_col2 tipo de dato);
```

En la parte que dice tipo de dato, esto se refiere a si es entero, caracteres y todo eso.

Ejemplo:



```
CREATE TABLE Persons (
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
);
```

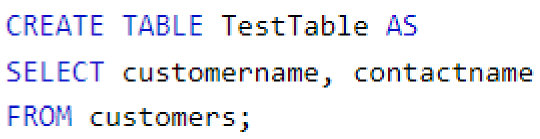
- **Crear una tabla con otra tabla**

Para crear una tabla en SQL, en base a otra tabla necesitaremos de otro SINTAXIS, siendo este de:

```
CREATE TABLE (nombre_tabla_nueva) AS
          SELECT columna_1,columna_2,...
          FROM tabla_existente
          WHERE …;
```

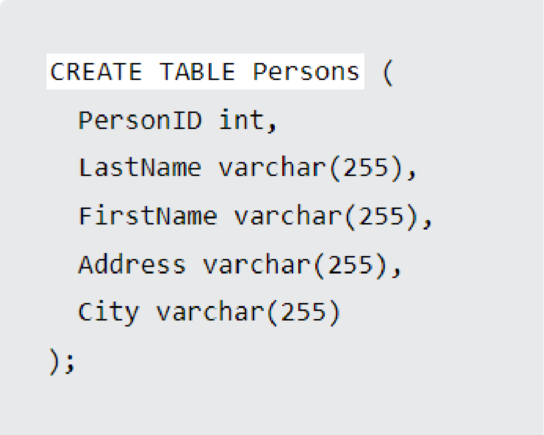
Siendo la tabla existente, la tabla en la que nos basaremos para crear la nueva tabla.

Ejemplo:

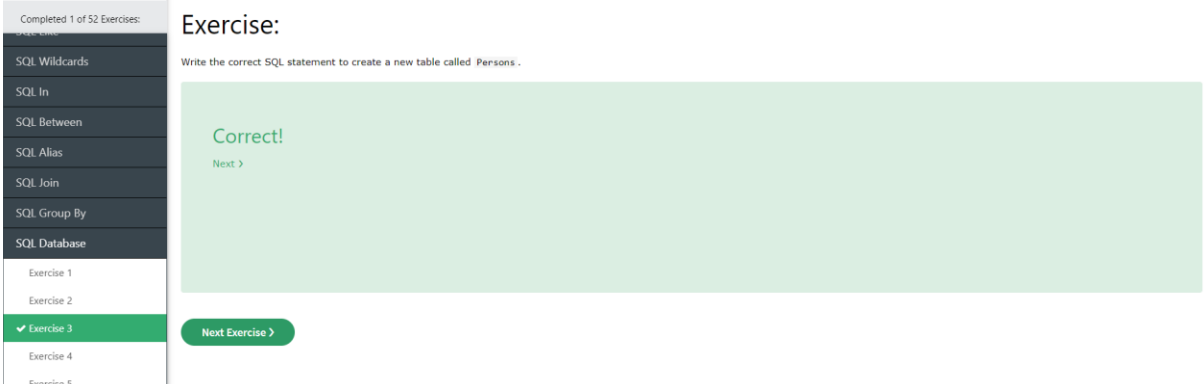


```
CREATE TABLE TestTable AS
SELECT customername, contactname
FROM customers;
```

EJERCICIO PRÁCTICO



```
CREATE TABLE Persons (
  PersonID int,
  LastName varchar(255),
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255)
);
```



Exercise:

Write the correct SQL statement to create a new table called Persons .

Correct!

Next ›

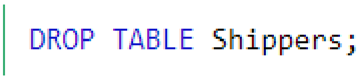
SQL DROP

- **DROP TABLE**

DROP TABLE , se usa para eliminar una tabla existente en una base de datos, la SINTAXIS es:

DROP TABLE (nombre_tabla);

Ejemplo:



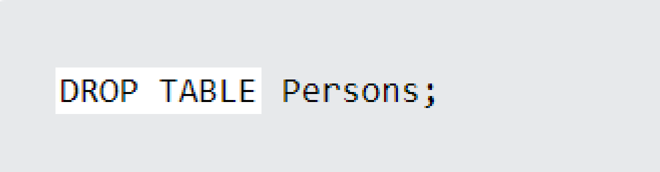
```
DROP TABLE Shippers;
```

- **TRUNCATE TABLE**

TRUNCATE TABLE se usa para eliminar los datos dentro de una tabla, pero no la tabla, su SINTAXIS es:

TRUNCATE TABLE (nombre_tabla);

EJERCICIO PRÁCTICO



```
DROP TABLE Persons;
```

Write the correct SQL statement to delete a table called `Persons` .

Correct!

SQL ALTER TABLE

- **ALTER TABLE**

Se utiliza para agregar, eliminar o modificar columnas en una tabla existente.
Se utiliza para agregar y eliminar varias restricciones en una tabla existente.

Agregar Columna a una tabla (SINTAXIS)

```
ALTER TABLE (nombre_tabla)
ADD (nombre_columna) (Tipo_dato);
```

Ejemplo:

```
ALTER TABLE Customers
ADD Email varchar(255);
```

Eliminar una columna de una tabla (SINTAXIS)

```
ALTER TABLE (table_name)
DROP COLUMN (nombre_columna);
```

Ejemplo:

```
ALTER TABLE Customers
DROP COLUMN Email;
```

Cambiar el nombre de una columna en una tabla (SINTAXIS)

```
ALTER TABLE (nombre_tabla)
RENAME COLUMN (nombre_antiguo) to (nuevo_nombre);
```

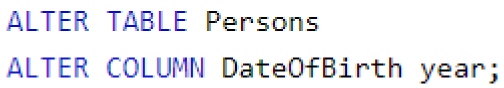
Si es una tabla en SQL SERVER, se usa la siguiente SINTAXIS:

```
EXEC sp_rename “nombre_tabla.nombre_antiguo”, “nuevo_nombre”, “COLUMN”;
```

Modificar tipo de datos (SINTAXIS)

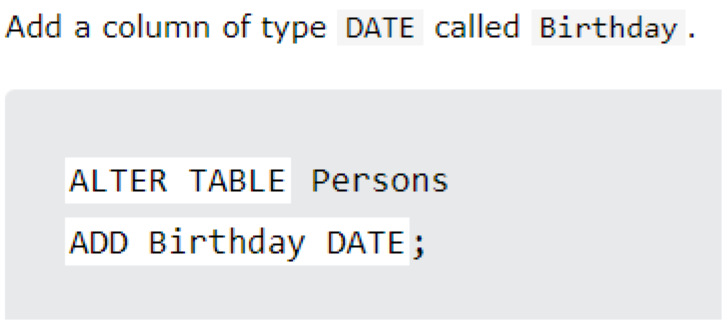
```
ALTER TABLE (nombre_tabla)
ALTER COLUMN (nombre_columna) (tipo de dato);
```

Ejemplo:



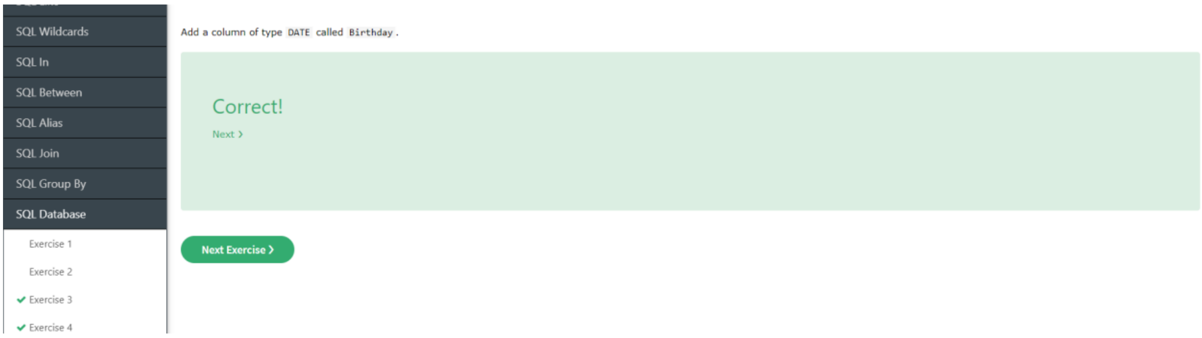
```
ALTER TABLE Persons
ALTER COLUMN DateOfBirth year;
```

EJEMPLO PRACTICO



Add a column of type DATE called Birthday.

```
ALTER TABLE Persons
ADD Birthday DATE;
```



Add a column of type DATE called Birthday.

Correct!

SQL CONSTRAINS

Una tabla puede contener restricciones cuando se crea, o incluso cuando se desea modificar.La SINTAXIS es:

```
CREATE TABLE (nombre_tabla) (
                    columna_1  tipo_dato  restricción,
                    columna_2  tipo_dato  restriccion,
                    columna_3  tipo_dato  restriccion,
                    …
);
```

Las restricciones de SQL, se utilizan para limitar el tipo de datos que se pueden incluir en una tabla. Las restricciones pueden ser a nivel de columna o de la tabla.

RESTRICCIONES COMUNES en SQL:

NOT NULL: Asegura que una columna no pueda tener un valor vacío (NULL).
UNIQUE: Asegura que todos los valores de una columna son diferentes.
PRIMARY KEY: Es una combinación de NOT NULL y UNIQUE. Donde identifica de forma única cada tabla de una tabla.
FOREIGN KEY: Evita que diversas acciones destruyan vínculos entre tablas.
CHECK: Asegura que los valores de una columna satisfacen una condición específica

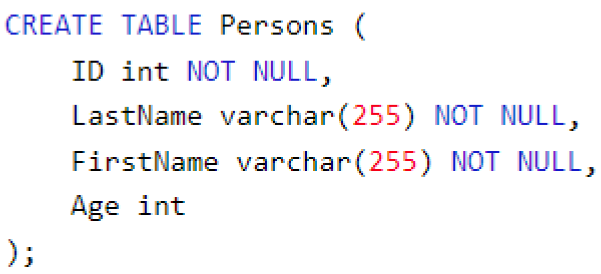
DEFAULT: Establece un valor predeterminado para una columna si no se especifica un valor
CREATE INDEX: Se utiliza para crear y recuperar los datos de la base de datos muy rápidamente

SQL NOT NULL

Una columna puede contener valores NULL, por esto mismo, la restricción NOT NULL obliga a una columna a NO aceptar valores NULOS.

- **NOT NULL EN CREAR TABLA (EJEMPLO)**

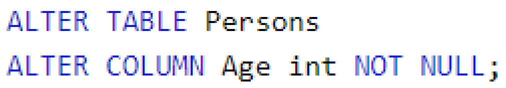
Este ejemplo lo que hace es crear una tabla, donde ni el ID,ni el primer y segundo nombre esten vacios.



```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255) NOT NULL,
    Age int
);
```

- **NOT NULL EN AGREGAR COLUMNA (EJEMPLO)**

Este ejemplo añade una columna Año, con el tipo de dato entero, donde este no puede estar vacío.

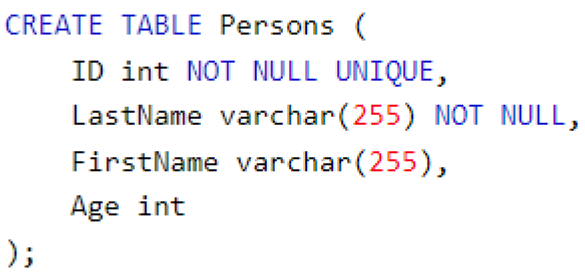


```
ALTER TABLE Persons
ALTER COLUMN Age int NOT NULL;
```

SQL UNIQUE

La restricción UNIQUE asegura que todos los valores de la columna son diferentes.

EJEMPLO:
En este ejemplo se crea una tabla con el nombre de personas, donde cada una tiene un ID que no es vacio, y este ID es diferente, tambien tiene una columna de primer nombre donde la cantidad de bits de la cadena de caractereses de 255 y no es vacia, segundo nombre donde la cantidad de bits de la cadena de caracteres es de 255 y edad



```
CREATE TABLE Persons (
    ID int NOT NULL UNIQUE,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int
);
```

Para nombrar una UNIQUE o definir una UNIQUE en varias columnas, se usa la siguiente SINTAXIS.

```
CREATE TABLE Persons(
        ID int NOT NULL,
        Lastname varchar(255) NOT NULL,
        Fistname varchar(255),
        Age int,
        CONSTRAINT UC_Person UNIQUE (ID,Lastname)
);
```

CREAR UNA RESTRICCIÓN UNIQUE CUANDO LA TABLA YA ESTÁ CREADA (SINTAXIS).

```
ALTER TABLE Persons
ADD UNIQUE (ID);
```

NOMBRAR UNA RESTRICCIÓN UNIQUE Y DEFINIRLA EN VARIAS COLUMNAS (SINTAXIS).

```
ALTER TABLE Persons
ADD CONSTRAINT UC_Person UNIQUE (ID,LastName);
```

ELIMINAR UNA RESTRICCIÓN ÚNICA:

```
ALTER TABLE Persons
DROP INDEX UC_Person;
```

SQL PRIMARY KEY

-La restricción PRIMARY KEY identifica de forma única cada registro de una tabla.

-Una clave primaria debe contener valores ÚNICOS y no puede contener valores NULOS

-Una tabla sólo puede contener una clave principal, esta clave principal puede constar en una o varias columnas.

CLAVE PRINCIPAL AL CREAR UNA TABLA (SINTAXIS)

```
CREATE TABLE Persons (
        ID int NOT NULL PRIMARY KEY ,
        Lastname varchar(255) NOT NULL,
        Firstname varchar(255),
        Age int
);
```

CLAVE PRINCIPAL CUANDO LA TABLA YA ESTÁ CREADA (SINTAXSIS)

En una columna:

```
ALTER TABLE Persons
ADD PRIMARY KEY(ID)
```

En varias columnas:

```
ALTER TABLE Persons
```

```
ADD CONSTRAINT PK_Person PRIMARY KEY (ID,LastName);
```

ELIMINAR UNA RESTRICCIÓN DE PRIMARY KEY.

```
ALTER TABLE Persons
DROP CONSTRAINT PK_Person;
```

SQL FOREIGN KEY

-La restricción FOREIGN KEY se utiliza para evitar acciones que podrían destruir vínculos entre tablas.
-La FOREIGN KEY es un campo o colección de estos en una tabla que hace referencia a otra tabla.
-La tabla con clave externa se denomina tabla secundaria y la tabla con clave principal se denomina tabla referenciada o tabla principal.

CREAR UNA TABLA ASIGNÁNDOLE UNA FOREIGN KEY.

```
CREATE TABLE Orders (
        OrderID int NOT NULL PRIMARY KEY,
        OrderNumber int NOT NULL,
        Name_col int FOREIGN KEY REFERENCES Table_name_pk(Name_col_pk)
```

AGREGAR A UNA COLUMNA DE UNA TABLA CREADA FOREIGN KEY.

```
ALTER TABLE Orders
ADD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);
```

AGREGAR A VARIAS COLUMNAS DE UNA TABLA CREADA FOREIGN KEY.

```
ALTER TABLE Orders
ADD CONSTRAINT FK_PersonOrder
FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);
```

ELIMINAR UNA FOREIGN KEY.

```
ALTER TABLE Orders
DROP CONSTRINT FK_PersonOrder;
```

SQL CHECK

-La restricción CHECK se utiliza para limitar el rango de valores que se puede colocar en una columna
-Si se define CHECK en una columna, solo se permitirán ciertos valores para esa columna.

CHECK CUANDO SE CREA UNA TABLA

```
CREATE TABLE Persons (
        ID int NOT NULL ,
        Lastname varchar(255) NOT NULL,
        Firstname varchar(255),
        Age int CHECK (Age>=18),
);
```

CHECK EN MÚLTIPLES COLUMNAS.

```
CREATE TABLE Persons (
          ID int NOT NULL ,
          Lastname varchar(255) NOT NULL,
          Firstname varchar(255),
          Age int CHECK (Age>=18),
          City varchar(255),
          CONSTRAINT CHK_Person CHECK (AGE>=18 AND City=”Sandnes”)
);
```

AGREGAR CHECK A UNA COLUMNA CUANDO LA TABLA YA ESTA CREADA

```
ALTER TABLE Persons
ADD CHECK (Age>=18);
```

AGREGAR CHECK A VARIAS COLUMNAS CUANDO LA TABLA YA ESTA CREADA

```
ALTER TABLE Persons
ADD CONSTRAINT CHK_PersonAge CHECK (Age>=18 AND City=”Sandnes”);
```

ELIMINAR UNA RESTRICCIÓN CHECK

```
ALTER TABLE Persons
DROP CONSTRAINT CHK_PersonAge;
```

SQL DEFAULT

La restricción DEFAULT se usa para establecer un valor predeterminado para una columna. El valor predeterminado se agregará a todos los registros nuevos, si no se especifica otro valor.

DEFAULT CUANDO SE CREA UNA TABLA

```
CREATE TABLE Persons (
          ID int NOT NULL ,
          Lastname varchar(255) NOT NULL,
          Firstname varchar(255),
          Age int,
          City varchar(255) DEFAULT “Sandnes”
);
```

NOTA: Cabe resaltar que en vez de "Sandnes" tambien se podrian colocar valores del sistema, como por ejemplo GETDATE()

DEFAULT CUANDO LA TABLA YA ESTÁ CREADA

```
ALTER TABLE Persons
ADD CONSTRINT df_City
DEFAULT "Sandnes" FOR City
```

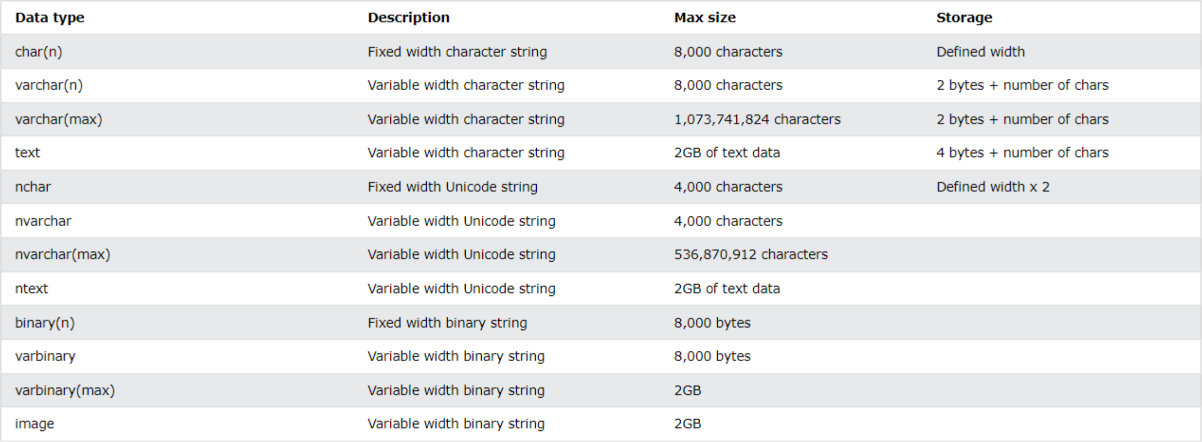
ELIMINAR UNA RESTRICCION DE DEFAULT

```
ALTER TABLE Persons
ALTER COLUMN City DROP DEFAULT;
```

SQL DATA TYPES

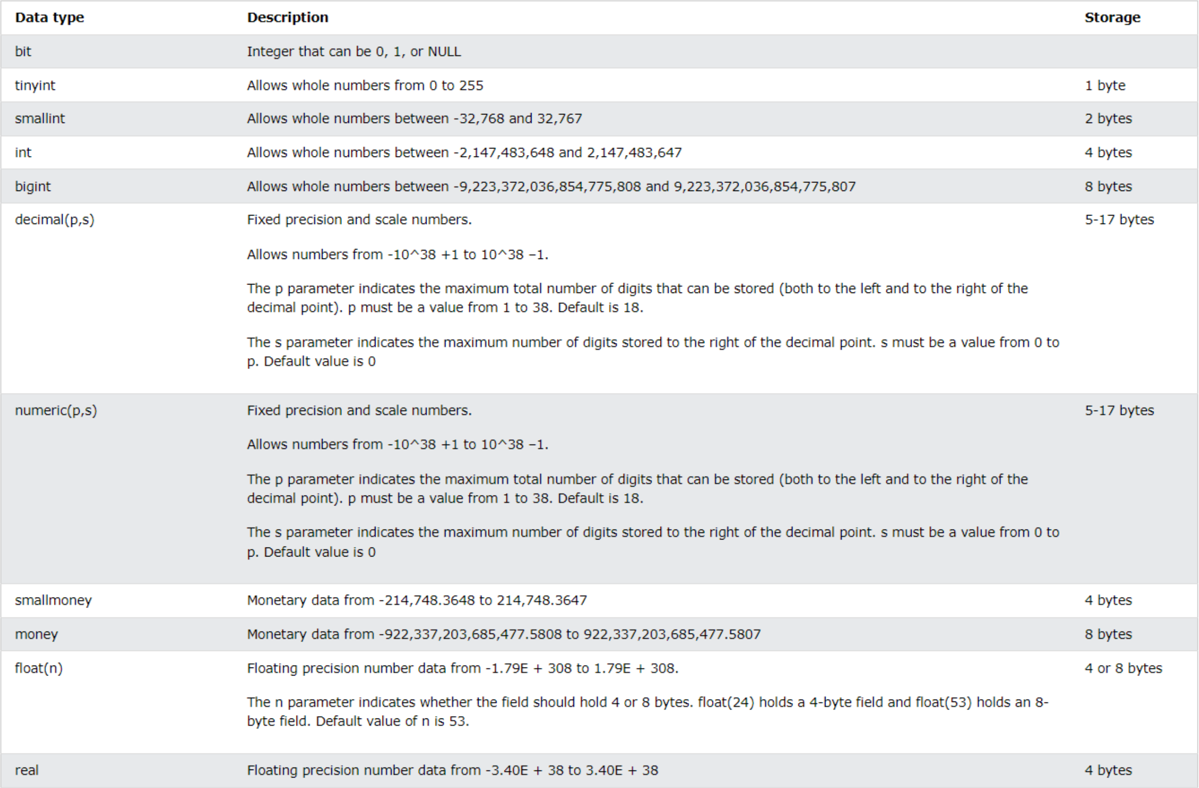
Cada columna de una tabla de bases de datos debe tener un nombre y un tipo de datos.

TIPOS DE DATOS DE CADENA



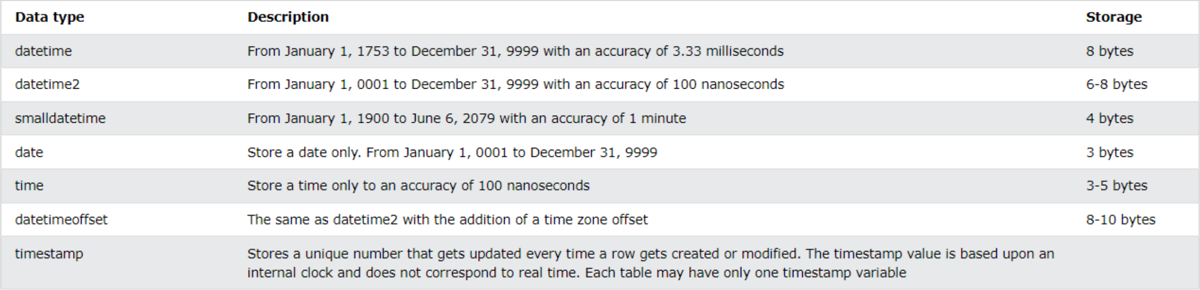
Data type	Description	Max size	Storage
char(n)	Fixed width character string	8,000 characters	Defined width
varchar(n)	Variable width character string	8,000 characters	2 bytes + number of chars
varchar(max)	Variable width character string	1,073,741,824 characters	2 bytes + number of chars
text	Variable width character string	2GB of text data	4 bytes + number of chars
nchar	Fixed width Unicode string	4,000 characters	Defined width x 2
nvarchar	Variable width Unicode string	4,000 characters	
nvarchar(max)	Variable width Unicode string	536,870,912 characters	
ntext	Variable width Unicode string	2GB of text data	
binary(n)	Fixed width binary string	8,000 bytes	
varbinary	Variable width binary string	8,000 bytes	
varbinary(max)	Variable width binary string	2GB	
image	Variable width binary string	2GB	

TIPOS DE DATOS NUMÉRICOS



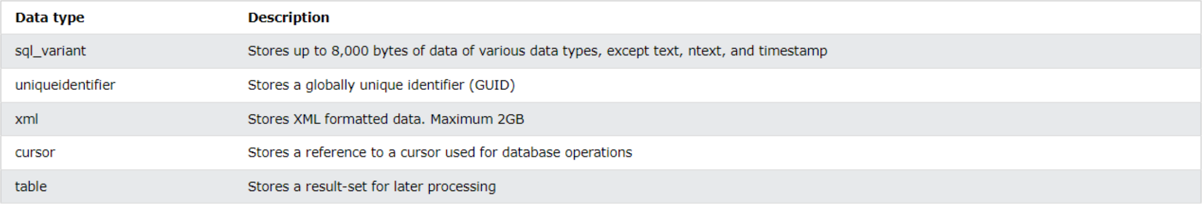
Data type	Description	Storage
bit	Integer that can be 0, 1, or NULL	
tinyint	Allows whole numbers from 0 to 255	1 byte
smallint	Allows whole numbers between -32,768 and 32,767	2 bytes
int	Allows whole numbers between -2,147,483,648 and 2,147,483,647	4 bytes
bigint	Allows whole numbers between -9,223,372,036,854,775,808 and 9,223,372,036,854,775,807	8 bytes
decimal(p,s)	Fixed precision and scale numbers. Allows numbers from -10^38 +1 to 10^38 –1. The p parameter indicates the maximum total number of digits that can be stored (both to the left and to the right of the decimal point). p must be a value from 1 to 38. Default is 18. The s parameter indicates the maximum number of digits stored to the right of the decimal point. s must be a value from 0 to p. Default value is 0	5-17 bytes
numeric(p,s)	Fixed precision and scale numbers. Allows numbers from -10^38 +1 to 10^38 –1. The p parameter indicates the maximum total number of digits that can be stored (both to the left and to the right of the decimal point). p must be a value from 1 to 38. Default is 18. The s parameter indicates the maximum number of digits stored to the right of the decimal point. s must be a value from 0 to p. Default value is 0	5-17 bytes
smallmoney	Monetary data from -214,748.3648 to 214,748.3647	4 bytes
money	Monetary data from -922,337,203,685,477.5808 to 922,337,203,685,477.5807	8 bytes
float(n)	Floating precision number data from -1.79E + 308 to 1.79E + 308. The n parameter indicates whether the field should hold 4 or 8 bytes. float(24) holds a 4-byte field and float(53) holds an 8-byte field. Default value of n is 53.	4 or 8 bytes
real	Floating precision number data from -3.40E + 38 to 3.40E + 38	4 bytes

TIPOS DE DATOS DE FECHA Y HORA



Data type	Description	Storage
datetime	From January 1, 1753 to December 31, 9999 with an accuracy of 3.33 milliseconds	8 bytes
datetime2	From January 1, 0001 to December 31, 9999 with an accuracy of 100 nanoseconds	6-8 bytes
smalldatetime	From January 1, 1900 to June 6, 2079 with an accuracy of 1 minute	4 bytes
date	Store a date only. From January 1, 0001 to December 31, 9999	3 bytes
time	Store a time only to an accuracy of 100 nanoseconds	3-5 bytes
datetimeoffset	The same as datetime2 with the addition of a time zone offset	8-10 bytes
timestamp	Stores a unique number that gets updated every time a row gets created or modified. The timestamp value is based upon an internal clock and does not correspond to real time. Each table may have only one timestamp variable	

OTROS TIPOS DE DATOS



Data type	Description
sql_variant	Stores up to 8,000 bytes of data of various data types, except text, ntext, and timestamp
uniqueidentifier	Stores a globally unique identifier (GUID)
xml	Stores XML formatted data. Maximum 2GB
cursor	Stores a reference to a cursor used for database operations
table	Stores a result-set for later processing

2. Completen la siguiente tabla de equivalencia de tipos de datos. Escriba los tipos usados en el modelo conceptual en SQL estandar y en SQL ORACLE.

Conceptual	SQL Estándar (SQL2)	SQL Oracle
Entero(n)	Integer o Numeric(n)	Number(n)
Real(D,M(	Real o Float(D,M)	Number(D,M) o Float

Caracter	**Char(1)**	**Char(1)**
Cadena(n) Fija	**Char(n)**	**Char(n)**
Cadena(n) Flexible	**Varchar(n) o Varchar2(n)**	**Varchar2(n)**
Hora	**Time**	**TIMESTAMP o DATE**
Fecha	**DATE**	**DATE**
Fecha+Hora	**Timestamp**	**Timestamp**

3. Practicando la definición de restricciones

A. Use la sentencia ALTER TABLE ADD CONSTRAINT para adicionar las restricciones de atributos y las claves de forma independiente a la creación de tablas. Revise el estándar de nombres. (Un ALTER TABLE por cada restricción)

CREATE TABLE AUTHOR(name VARCHAR(10) PRIMARY KEY, awards NUMBER(2) CHECK (awards>=0)) La restricción de llave primaria de debe llamar PK_AUTHOR La restricción de chequeo de tipo debe llamarse CK_AUTHOR_AWARDS	**CREATE TABLE AUTHOR(** **name VARCHAR(10) PRIMARY KEY** **awards NUMBER(2) CHECK (awards>=0)** **)** **ALTER TABLE AUTHOR** **ADD CONSTRAINT PK_AUTHOR PRIMARY KEY (name)** **ALTER TABLE AUTHOR** **ADD CONSTRAINT CK_AUTHOR_AWARDS CHECK (awards >= 0)**
CREATE TABLE SONG id NUMBER(5) PRIMARY KEY, name VARCHAR(30) NOT NULL UNIQUE, author VARCHAR(10) references AUTHOR(name)); La restricción de llave primaria se debe llamar PK_SONG La restricción de llave única se debe llamar FK_SONG_AUTHOR	**CREATE TABLE SONG(** **id NUMBER(5) PRIMARY KEY** **name VARCHAR(30) NOT NULL UNIQUE** **author VARCHAR(10) references AUTHOR(name)** **)** **ALTER TABLE SONG** **ADD CONSTRAINT PK_SONG PRIMARY KEY (id)** **ALTER TABLE SONG** **ADD CONSTRAINT UK_SONG_NAME UNIQUE(name)**

	ALTER TABLE SONG **ADD CONSTRAINT FK_SONG_AUTHOR FOREIGN KEY (author) REFERENCES AUTHOR(name)**

B. INVESTIGANDO SQL Developer

Considerando la herramienta SQL Developer

A. Investigue las funcionalidades básicas de la herramienta.

- Crear conexiones:
  Se pueden crear, establecer y probar conexiones a diferentes bases de datos externas a Oracle y otras bases de datos.
- Examinar objetos:
  Este navegador está basado en árbol, permite examinar y gestionar objetos de la base de datos como tablas, vistas, índices, procedimientos almacenados, funciones, tipos, secuencias, directorios, y más.
- Crear objetos:
  Los usuarios tienen la opción de configurar secuencias antes de la inserción para completar automáticamente una columna con valores, esto si se define una nueva tabla.
- Modificar objetos:
  La mayoría de objetos tienen opciones específicas de modificación accesibles a través del menú contextual al hacer clic derecho.
- Consultar y actualizar datos:
  Los usuarios pueden ejecutar consultas SQL para recuperar información de la base de datos y actualizar registros directamente desde la interfaz de SQL Developer.
- Exportar datos y DDL, importar datos:
  Esta funcionalidad permite la gestión y migración de datos en SQL Developer.
  Permite exportar datos de la base de datos a varios formatos, como CSV, XML, Excel, HTML y PDF.
- Migrar desde bases de datos de terceros:
  SQL Developer incluye herramientas robustas para facilitar la migración de datos y esquemas desde bases de datos de terceros a Oracle.

¿Qué es SQL Developer? | Oracle España

B. Indique sus ventajas y desventajas sobre otras herramientas similares.

- VENTAJAS:
  1. Gratuito: SQL Developer es gratis, por lo que cualquier usuario puede acceder sin invertir en licencias.
  2. Interfaz gráfica intuitiva: Esto hace que la navegación y gestión de bases de datos sea más fácil.
  3. Compatibilidad multiplataforma: funciona en cualquier sistema operativo compatible con Java.
  4. Amplias funcionalidades: Permite ejecutar consultas, depurar y probar scripts SQL y exportar datos a varios formatos.
     ¿Qué es SQL Developer? | Oracle España
- DESVENTAJAS:
  1. Rendimiento: Al manejar grandes volúmenes de datos o ejecutar consultas muy complejas, puede afectar la productividad y volverlo más lento.
  2. Curva de aprendizaje: A pesar de que la interfaz puede ser bastante intuitiva, existen funcionalidades avanzadas que requieren de tiempo para poder dominar.
  3. Limitaciones en bases de datos no oracle: Aunque soporta bases de datos de terceros,

las funcionalidades pueden ser limitadas.
[Comparativa: SQL Developer vs MySQL – MySQL YA]

Instalando

Instale la herramienta SQL Developer. ¿Son claras las instrucciones de instalación? ¿Se le presentó algún problema?

Para poder instalar el programa si fue necesario ver un par de videos porque tocaba instalar dos archivos para poder utilizar la aplicación, además que si es un poco pesada la instalación. Entonces no es que sea tan confuso el instalarlo pero toca saber bien cómo hacerlo.

De igual manera a la hora de utilizar el programa se tuvo que ver un tutorial para la creación de la base de datos, porque ya a la hora de crear las tablas y manipularlas fue más sencillo por la investigación del primer punto.

Arrancando

Realice y explique cómo se deben realizar las siguientes acciones:

- □ Establecer una conexión con el motor ORACLE de la ESCUELA
- □ Consultar toda la información posible que hay en su cuenta

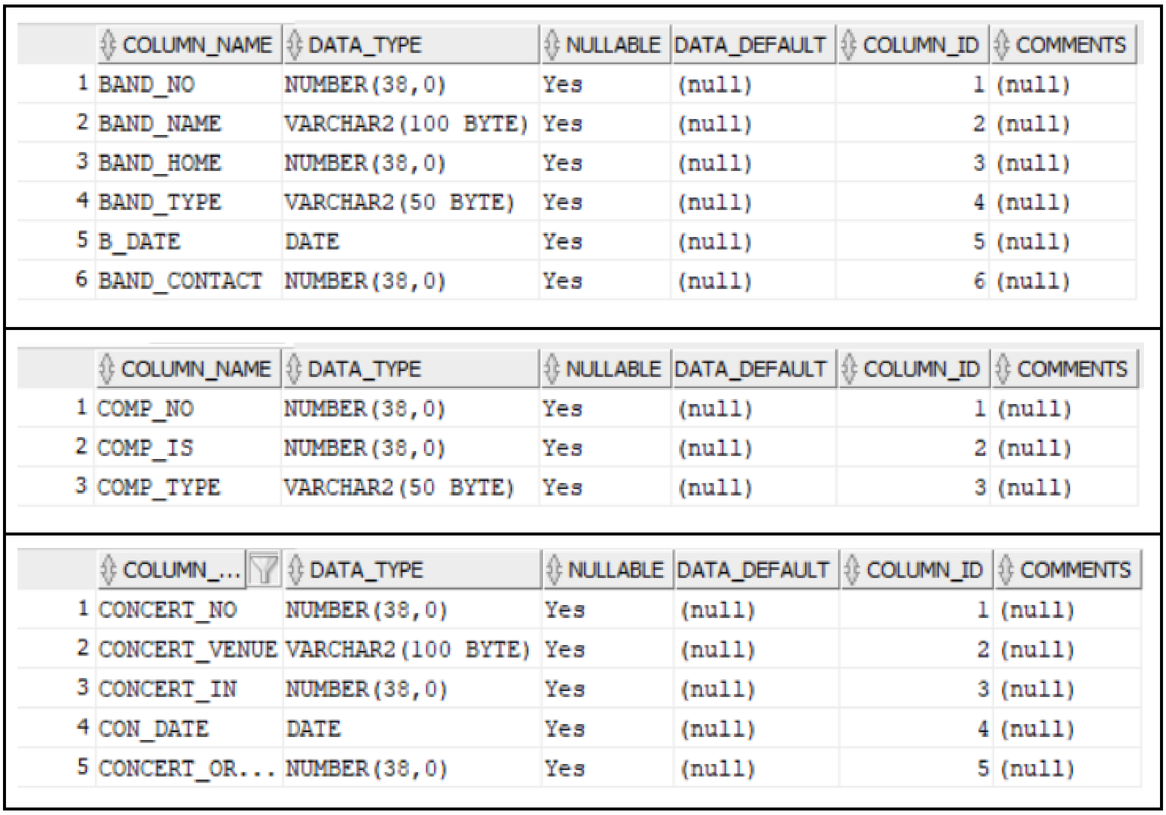
C. PRACTICANDO. GuestHouse

Para escribir el primer archivo de comandos .sql vamos a crear un subconjunto de la base de datos de **Musicians**: Ciclo Uno.

□ Crear la base de datos ciclo uno sin restricciones (Tablas)

```
CREATE TABLE band (
   band_no INT,
   band_name VARCHAR(100),
   band_home INT,
   band_type VARCHAR2(50),
   b_date DATE,
   band_contact INT
);

CREATE TABLE composer (
   comp_no INT,
   comp_is INT,
   comp_type VARCHAR2(50)
);


CREATE TABLE concert (
   concert_no INT,
   concert_venue VARCHAR2(100),
   concert_in INT,
   con_date DATE,
   concert_organiser INT
);

CREATE TABLE musician (
   m_no INT,
   m_name VARCHAR2(100),
   born DATE,
   died DATE,
   born_in INT,
   living_in INT
);

CREATE TABLE performance (
   pfmnce_no INT,
   gave INT,
   performed INT,
   conducted_by INT,
   performed_in INT
);

CREATE TABLE performer (
   perf_no INT,
   perf_is INT,
   instrument VARCHAR2(50),
   perf_type VARCHAR2(50)
);

CREATE TABLE place (
   place_no INT,
   place_town VARCHAR2(100),
   place_country VARCHAR2(100)
);

CREATE TABLE plays_in (
   player INT,
   band_id INT
);
```

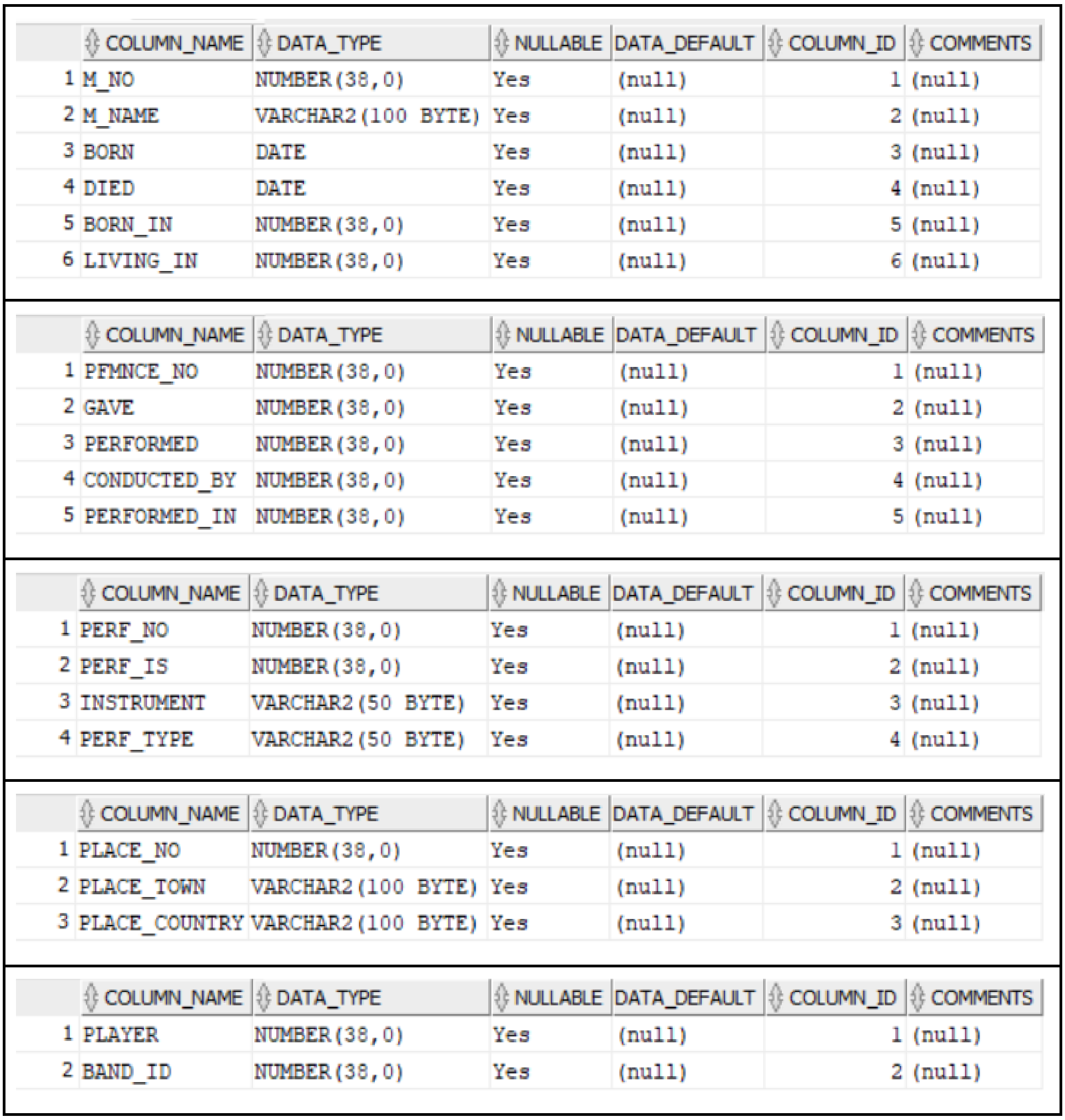
Estos fueron los comandos que se usaron a la hora de la creación de las tablas en Oracle. El resultado fue el siguiente:



	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	BAND_NO	NUMBER(38,0)	Yes	(null)	1	(null)
2	BAND_NAME	VARCHAR2(100 BYTE)	Yes	(null)	2	(null)
3	BAND_HOME	NUMBER(38,0)	Yes	(null)	3	(null)
4	BAND_TYPE	VARCHAR2(50 BYTE)	Yes	(null)	4	(null)
5	B_DATE	DATE	Yes	(null)	5	(null)
6	BAND_CONTACT	NUMBER(38,0)	Yes	(null)	6	(null)

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	COMP_NO	NUMBER(38,0)	Yes	(null)	1	(null)
2	COMP_IS	NUMBER(38,0)	Yes	(null)	2	(null)
3	COMP_TYPE	VARCHAR2(50 BYTE)	Yes	(null)	3	(null)

	COLUMN_...	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	CONCERT_NO	NUMBER(38,0)	Yes	(null)	1	(null)
2	CONCERT_VENUE	VARCHAR2(100 BYTE)	Yes	(null)	2	(null)
3	CONCERT_IN	NUMBER(38,0)	Yes	(null)	3	(null)
4	CON_DATE	DATE	Yes	(null)	4	(null)
5	CONCERT_OR...	NUMBER(38,0)	Yes	(null)	5	(null)



	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	M_NO	NUMBER(38,0)	Yes	(null)	1	(null)
2	M_NAME	VARCHAR2(100 BYTE)	Yes	(null)	2	(null)
3	BORN	DATE	Yes	(null)	3	(null)
4	DIED	DATE	Yes	(null)	4	(null)
5	BORN_IN	NUMBER(38,0)	Yes	(null)	5	(null)
6	LIVING_IN	NUMBER(38,0)	Yes	(null)	6	(null)

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	PFMNCE_NO	NUMBER(38,0)	Yes	(null)	1	(null)
2	GAVE	NUMBER(38,0)	Yes	(null)	2	(null)
3	PERFORMED	NUMBER(38,0)	Yes	(null)	3	(null)
4	CONDUCTED_BY	NUMBER(38,0)	Yes	(null)	4	(null)
5	PERFORMED_IN	NUMBER(38,0)	Yes	(null)	5	(null)

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	PERF_NO	NUMBER(38,0)	Yes	(null)	1	(null)
2	PERF_IS	NUMBER(38,0)	Yes	(null)	2	(null)
3	INSTRUMENT	VARCHAR2(50 BYTE)	Yes	(null)	3	(null)
4	PERF_TYPE	VARCHAR2(50 BYTE)	Yes	(null)	4	(null)

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	PLACE_NO	NUMBER(38,0)	Yes	(null)	1	(null)
2	PLACE_TOWN	VARCHAR2(100 BYTE)	Yes	(null)	2	(null)
3	PLACE_COUNTRY	VARCHAR2(100 BYTE)	Yes	(null)	3	(null)

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	PLAYER	NUMBER(38,0)	Yes	(null)	1	(null)
2	BAND_ID	NUMBER(38,0)	Yes	(null)	2	(null)

□ Adicionar las restricciones declarativas a la base de datos
(Atributos, Primarias, Únicas, Foraneas)

ALTER TABLE MUSICIAN
ADD CONSTRAINT PK_MUSICIAN PRIMARY KEY (m_no);

ALTER TABLE BAND
ADD CONSTRAINT PK_BAND PRIMARY KEY (band_no);

ALTER TABLE COMPOSER
ADD CONSTRAINT PK_COMPOSER PRIMARY KEY (comp_no);

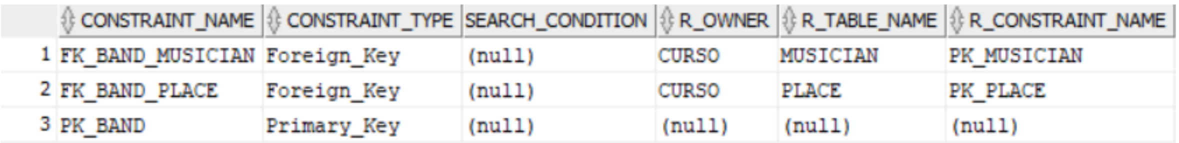
ALTER TABLE CONCERT
ADD CONSTRAINT PK_CONCERT PRIMARY KEY (concert_no);

ALTER TABLE PERFORMANCE

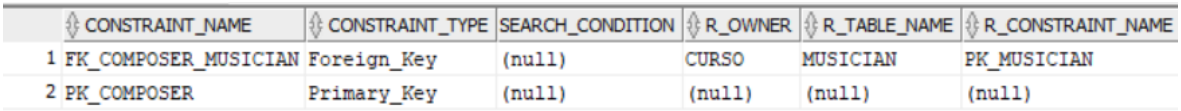
```
ADD CONSTRAINT PK_PERFORMANCE PRIMARY KEY (pfmnce_no);

ALTER TABLE PERFORMER
ADD CONSTRAINT PK_PERFORMER PRIMARY KEY (perf_no);

ALTER TABLE PLACE
ADD CONSTRAINT PK_PLACE PRIMARY KEY (place_no);

ALTER TABLE PLAYS_IN
ADD CONSTRAINT PK1_PLAY PRIMARY KEY (player, band_id);

ALTER TABLE PLAYS_IN
ADD CONSTRAINT FK_PERFORMER_PLAY FOREIGN KEY (player) references
performer(perf_no)
ADD CONSTRAINT FK_BAND_PLAY FOREIGN KEY (band_id) references band(band_no);


ALTER TABLE MUSICIAN
ADD CONSTRAINT FK1_MUSICIAN_PLACE FOREIGN KEY (born_in) references
place(place_no)
ADD CONSTRAINT FK2_MUSICIAN_PLACE FOREIGN KEY (living_in) references
place(place_no);

ALTER TABLE CONCERT
ADD CONSTRAINT FK_CONCERT_PLACE FOREIGN KEY (concert_in) references
place(place_no);

ALTER TABLE PERFORMANCE
ADD CONSTRAINT FK_PERFORMANCE_PLACE FOREIGN KEY (performed_in)
references place(place_no);

ALTER TABLE BAND
ADD CONSTRAINT FK_BAND_PLACE FOREIGN KEY (band_home) references
place(place_no);

ALTER TABLE PERFORMANCE
ADD CONSTRAINT FK_PERFORMANCE_BAND FOREIGN KEY (gave) references
band(band_no)
ADD CONSTRAINT FK_PERFORMANCE_MUSICIAN FOREIGN KEY (conducted_by)
references musician(m_no);

ALTER TABLE COMPOSER
ADD CONSTRAINT FK_COMPOSER_MUSICIAN FOREIGN KEY (comp_is) references
musician(m_no);

ALTER TABLE PERFORMER
ADD CONSTRAINT FK_PERFORMER_MUSICIAN FOREIGN KEY (perf_is) references
musician(m_no);

ALTER TABLE CONCERT
ADD CONSTRAINT FK_CONCERT_MUSICIAN FOREIGN KEY (concert_organiser)
references musician(m_no);

ALTER TABLE BAND
ADD CONSTRAINT FK_BAND_MUSICIAN FOREIGN KEY (band_contact) references
```

musician(m_no);

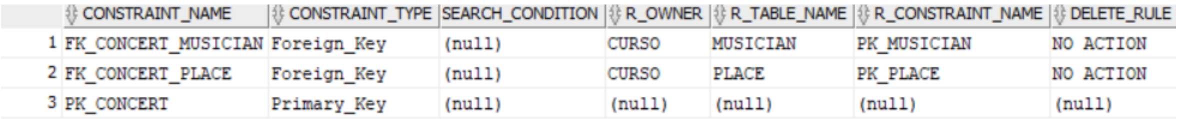
Las restricciones en las tablas quedaron de la siguiente manera:



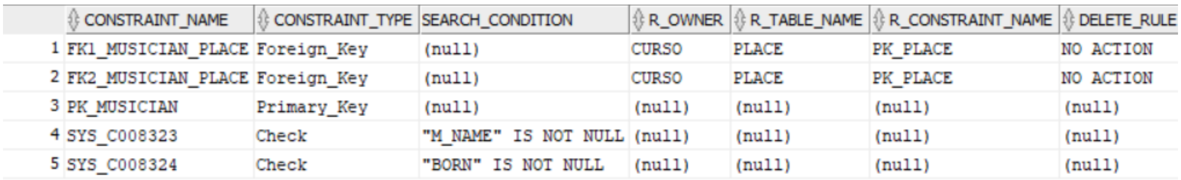
	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_OWNER	R_TABLE_NAME	R_CONSTRAINT_NAME
1	FK_BAND_MUSICIAN	Foreign_Key	(null)	CURSO	MUSICIAN	PK_MUSICIAN
2	FK_BAND_PLACE	Foreign_Key	(null)	CURSO	PLACE	PK_PLACE
3	PK_BAND	Primary_Key	(null)	(null)	(null)	(null)



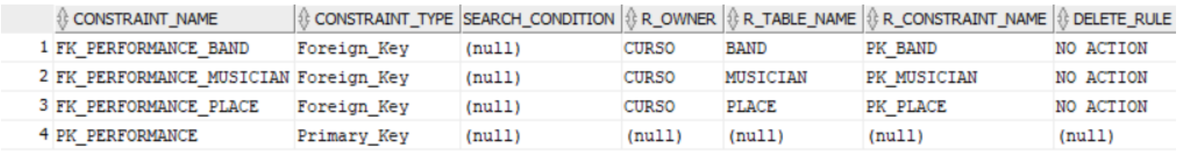
	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_OWNER	R_TABLE_NAME	R_CONSTRAINT_NAME
1	FK_COMPOSER_MUSICIAN	Foreign_Key	(null)	CURSO	MUSICIAN	PK_MUSICIAN
2	PK_COMPOSER	Primary_Key	(null)	(null)	(null)	(null)



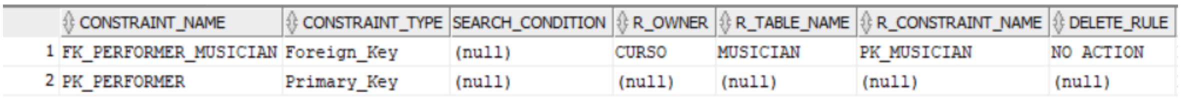
	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_OWNER	R_TABLE_NAME	R_CONSTRAINT_NAME	DELETE_RULE
1	FK_CONCERT_MUSICIAN	Foreign_Key	(null)	CURSO	MUSICIAN	PK_MUSICIAN	NO ACTION
2	FK_CONCERT_PLACE	Foreign_Key	(null)	CURSO	PLACE	PK_PLACE	NO ACTION
3	PK_CONCERT	Primary_Key	(null)	(null)	(null)	(null)	(null)



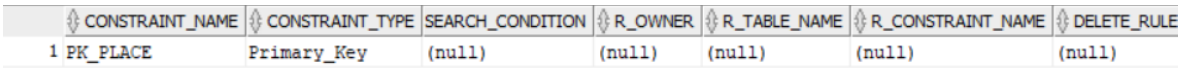
	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_OWNER	R_TABLE_NAME	R_CONSTRAINT_NAME	DELETE_RULE
1	FK1_MUSICIAN_PLACE	Foreign_Key	(null)	CURSO	PLACE	PK_PLACE	NO ACTION
2	FK2_MUSICIAN_PLACE	Foreign_Key	(null)	CURSO	PLACE	PK_PLACE	NO ACTION
3	PK_MUSICIAN	Primary_Key	(null)	(null)	(null)	(null)	(null)
4	SYS_C008323	Check	"M_NAME" IS NOT NULL	(null)	(null)	(null)	(null)
5	SYS_C008324	Check	"BORN" IS NOT NULL	(null)	(null)	(null)	(null)



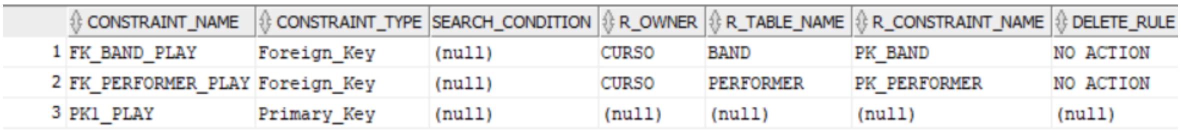
	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_OWNER	R_TABLE_NAME	R_CONSTRAINT_NAME	DELETE_RULE
1	FK_PERFORMANCE_BAND	Foreign_Key	(null)	CURSO	BAND	PK_BAND	NO ACTION
2	FK_PERFORMANCE_MUSICIAN	Foreign_Key	(null)	CURSO	MUSICIAN	PK_MUSICIAN	NO ACTION
3	FK_PERFORMANCE_PLACE	Foreign_Key	(null)	CURSO	PLACE	PK_PLACE	NO ACTION
4	PK_PERFORMANCE	Primary_Key	(null)	(null)	(null)	(null)	(null)



	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_OWNER	R_TABLE_NAME	R_CONSTRAINT_NAME	DELETE_RULE
1	FK_PERFORMER_MUSICIAN	Foreign_Key	(null)	CURSO	MUSICIAN	PK_MUSICIAN	NO ACTION
2	PK_PERFORMER	Primary_Key	(null)	(null)	(null)	(null)	(null)



	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_OWNER	R_TABLE_NAME	R_CONSTRAINT_NAME	DELETE_RULE
1	PK_PLACE	Primary_Key	(null)	(null)	(null)	(null)	(null)



	CONSTRAINT_NAME	CONSTRAINT_TYPE	SEARCH_CONDITION	R_OWNER	R_TABLE_NAME	R_CONSTRAINT_NAME	DELETE_RULE
1	FK_BAND_PLAY	Foreign_Key	(null)	CURSO	BAND	PK_BAND	NO ACTION
2	FK_PERFORMER_PLAY	Foreign_Key	(null)	CURSO	PERFORMER	PK_PERFORMER	NO ACTION
3	PK1_PLAY	Primary_Key	(null)	(null)	(null)	(null)	(null)

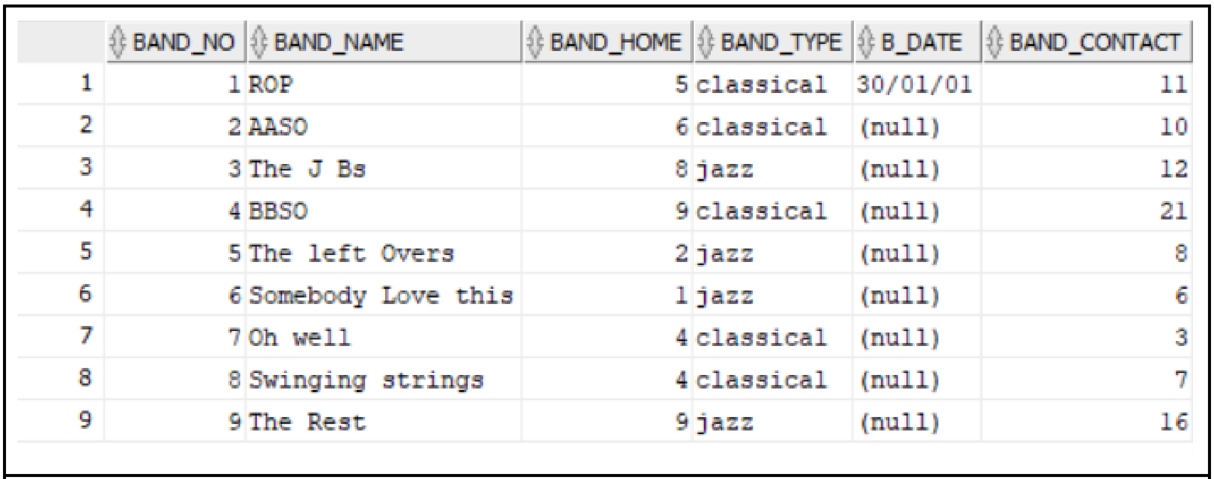
□ Poblar la base de datos con los datos iniciales (PoblarOK)
Automaticen la generación de las instrucciones INSERT.
Dejen en el archivo las consultas correspondientes en comentarios.

```
INSERT INTO PLACE(place_no, place_town, place_country) VALUES (1, 'Manchester',
'England');
INSERT INTO PLACE(place_no, place_town, place_country) VALUES (2, 'Edinburgh',
'Scotland');
INSERT INTO PLACE(place_no, place_town, place_country) VALUES (3, 'Salzburg',
'Austria');
INSERT INTO PLACE(place_no, place_town, place_country) VALUES (4, 'New York', 'USA');
INSERT INTO PLACE(place_no, place_town, place_country) VALUES (5, 'Birmingham',
'England');
INSERT INTO PLACE(place_no, place_town, place_country) VALUES (6, 'Glasglow',
'Scotland');
INSERT INTO PLACE(place_no, place_town, place_country) VALUES (7, 'London',
'England');
INSERT INTO PLACE(place_no, place_town, place_country) VALUES (8, 'Chicago', 'USA');
INSERT INTO PLACE(place_no, place_town, place_country) VALUES (9, 'Amsterdam',
'Netherlands');

INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (1, 'Fred
Bloggs', '02/03/1948', NULL, 1, 2);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (2, 'John
Smith', '03/03/1950', NULL, 3, 4);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (3, 'Helen
Smyth', '08/08/48', NULL, 4, 5);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (4,
'Harriet Smithson', '09/05/1909', '20/09/80', 5 ,6);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (5, 'James
First', '10/06/1965', NULL, 7 ,7);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (6, 'Theo
Mengel', '12/08/1948', NULL, 7 ,1);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (7, 'Sue
Little', '21/02/1945', NULL, 8 ,9);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (8, 'Harry
Forte', '28/02/1951', NULL, 1 ,8);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (9, 'Phil
Hot', '30/06/1942', NULL, 2 ,7);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (10, 'Jeff
Dawn', '12/12/1945', NULL, 3 ,6);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (11, 'Rose
Spring', '25/05/1948', NULL, 4 ,5);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (12,
'Davis Heavan', '03/10/1975', NULL, 5 ,4);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (13,
'Lovely Time', '28/12/1948', NULL, 6 ,3);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (14, 'Alan
Fluff', '15/01/1935', '15/05/1997', 7 ,2);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (15,
'Tony Smythe', '02/04/1932', NULL, 8 ,1);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (16,
'James Quick', '08/08/1924', NULL, 9 ,2);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (17,
'Freda Miles', '04/07/1920', NULL, 9 ,3);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (18, 'Elsie
James', '06/05/1947', NULL, 8 ,5);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (19,
'Andy Jones', '08/10/1958', NULL, 7 ,6);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (20,
'Louise Simpson', '10/01/1948', '11/02/1998', 6 ,6);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (21,
'James Steeple', '10/01/1947', NULL, 5 ,6);
INSERT INTO MUSICIAN(m_no, m_name, born, died, born_in, living_in) VALUES (22,
'Steven Chaytors', '11/03/1956', NULL, 6 ,7);
```

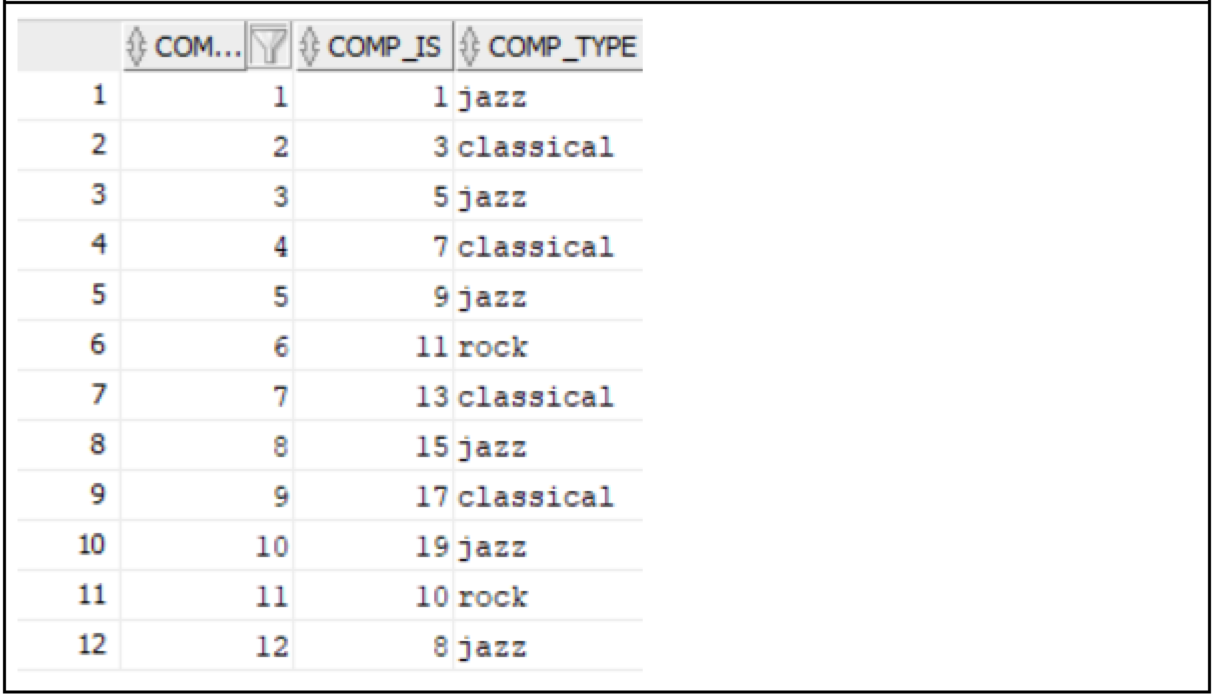
```
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, band_contact)
VALUES(1, 'ROP', 5, 'classical', '30/01/2001', 11);
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, band_contact)
VALUES(2, 'AASO', 6, 'classical', NULL, 10);
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, band_contact)
VALUES(3, 'The J Bs', 8, 'jazz', NULL, 12);
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, band_contact)
VALUES(4, 'BBSO', 9, 'classical', NULL, 21);
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, band_contact)
VALUES(5, 'The left Overs', 2, 'jazz', NULL, 8);
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, band_contact)
VALUES(6, 'Somebody Love this', 1, 'jazz', NULL, 6);
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, band_contact)
VALUES(7, 'Oh well', 4, 'classical', NULL, 3);
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, band_contact)
VALUES(8, 'Swinging strings', 4, 'classical', NULL, 7);
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, band_contact)
VALUES(9, 'The Rest', 9, 'jazz', NULL, 16);

INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(1,1,'jazz');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(2,3,'classical');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(3,5,'jazz');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(4,7,'classical');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(5,9,'jazz');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(6,11,'rock');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(7,13,'classical');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(8,15,'jazz');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(9,17,'classical');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(10,19,'jazz');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(11,10,'rock');
INSERT INTO COMPOSER(comp_no, comp_is, comp_type) VALUES(12,8,'jazz');


INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(1,2,'violin','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(2,4,'viola','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(3,6,'banjo','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(4,8,'violin','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(5,12,'guitar','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(6,14,'violin','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(7,16,'trumpet','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(8,18,'viola','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(9,20,'bass','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(10,2,'flute','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(11,20,'cornet','jazz');
```

```
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(12,6,'violin','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(13,8,'drums','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(14,10,'violin','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(15,12,'cello','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(16,14,'viola','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(17,16,'flute','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(18,18,'guitar','not known');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(19,20,'trombone','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(20,3,'horn','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(21,5,'violin','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(22,7,'cello','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(23,2,'bass','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(24,4,'violin','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(25,6,'drums','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(26,8,'clarinet','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(27,10,'bass','jazz');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(28,12,'viola','classical');
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type)
VALUES(29,18,'cello','classical');


INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date,
concert_organiser) VALUES (1, 'Bridgewater Hall', 1, '06/01/1995', 21);
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date,
concert_organiser) VALUES (2, 'Bridgewater Hall', 1,'08/05/1996', 3);
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date,
concert_organiser) VALUES (3, 'Usher Hall', 2, '03/06/1995', 3);
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date,
concert_organiser) VALUES (4, 'Assembly Rooms', 2, '20/09/1997', 21);
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date,
concert_organiser) VALUES (5, 'Festspiel Haus', 3, '21/02/1995', 8);
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date,
concert_organiser) VALUES (6, 'Royal Albert Hall', 7, '12/04/1993', 8);
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date,
concert_organiser) VALUES (7, 'Concertgebouw', 9, '14/05/1993', 8);
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date,
concert_organiser) VALUES (8, 'Metropolitan', 4, '15/06/1997', 21);
```

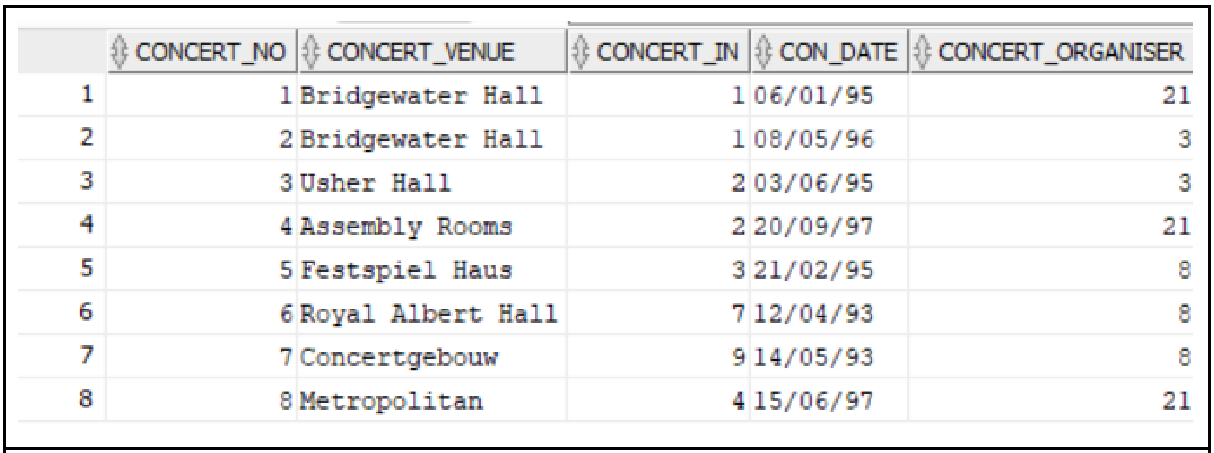
```
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(1,1,1,21,1);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(2,1,3,21,1);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(3,1,5,21,1);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(4,1,2,1,2);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(5,2,4,21,2);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(6,2,6,21,2);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(7,4,19,9,3);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(8,4,20,10,3);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(9,5,12,10,4);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(10,5,13,11,4);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(11,3,5,13,5);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(12,3,6,13,5);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(13,3,7,13,5);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(14,6,20,14,6);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(15,8,12,15,7);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(16,9,16,21,8);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(17,9,17,21,8);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(18,9,18,21,8);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(19,9,19,21,8);
INSERT INTO PERFORMANCE(pfmnce_no, gave, performed, conducted_by, performed_in)
VALUES(20,4,12,10,3);

INSERT INTO PLAYS_IN(player, band_id) VALUES(1,1);
INSERT INTO PLAYS_IN(player, band_id) VALUES(1,7);
INSERT INTO PLAYS_IN(player, band_id) VALUES(3,1);
INSERT INTO PLAYS_IN(player, band_id) VALUES(4,1);
INSERT INTO PLAYS_IN(player, band_id) VALUES(4,7);
INSERT INTO PLAYS_IN(player, band_id) VALUES(5,1);
INSERT INTO PLAYS_IN(player, band_id) VALUES(6,1);
INSERT INTO PLAYS_IN(player, band_id) VALUES(6,7);
INSERT INTO PLAYS_IN(player, band_id) VALUES(7,1);
INSERT INTO PLAYS_IN(player, band_id) VALUES(8,1);
INSERT INTO PLAYS_IN(player, band_id) VALUES(8,7);
INSERT INTO PLAYS_IN(player, band_id) VALUES(10,2);
INSERT INTO PLAYS_IN(player, band_id) VALUES(12,2);
INSERT INTO PLAYS_IN(player, band_id) VALUES(13,2);
INSERT INTO PLAYS_IN(player, band_id) VALUES(14,2);
INSERT INTO PLAYS_IN(player, band_id) VALUES(14,8);
INSERT INTO PLAYS_IN(player, band_id) VALUES(15,2);
INSERT INTO PLAYS_IN(player, band_id) VALUES(15,8);
INSERT INTO PLAYS_IN(player, band_id) VALUES(17,2);
INSERT INTO PLAYS_IN(player, band_id) VALUES(18,2);
INSERT INTO PLAYS_IN(player, band_id) VALUES(19,3);
INSERT INTO PLAYS_IN(player, band_id) VALUES(20,3);
INSERT INTO PLAYS_IN(player, band_id) VALUES(21,4);
INSERT INTO PLAYS_IN(player, band_id) VALUES(22,4);
INSERT INTO PLAYS_IN(player, band_id) VALUES(23,4);
INSERT INTO PLAYS_IN(player, band_id) VALUES(25,5);
INSERT INTO PLAYS_IN(player, band_id) VALUES(26,6);
INSERT INTO PLAYS_IN(player, band_id) VALUES(27,6);
INSERT INTO PLAYS_IN(player, band_id) VALUES(28,7);
INSERT INTO PLAYS_IN(player, band_id) VALUES(28,8);
INSERT INTO PLAYS_IN(player, band_id) VALUES(29,7);
```



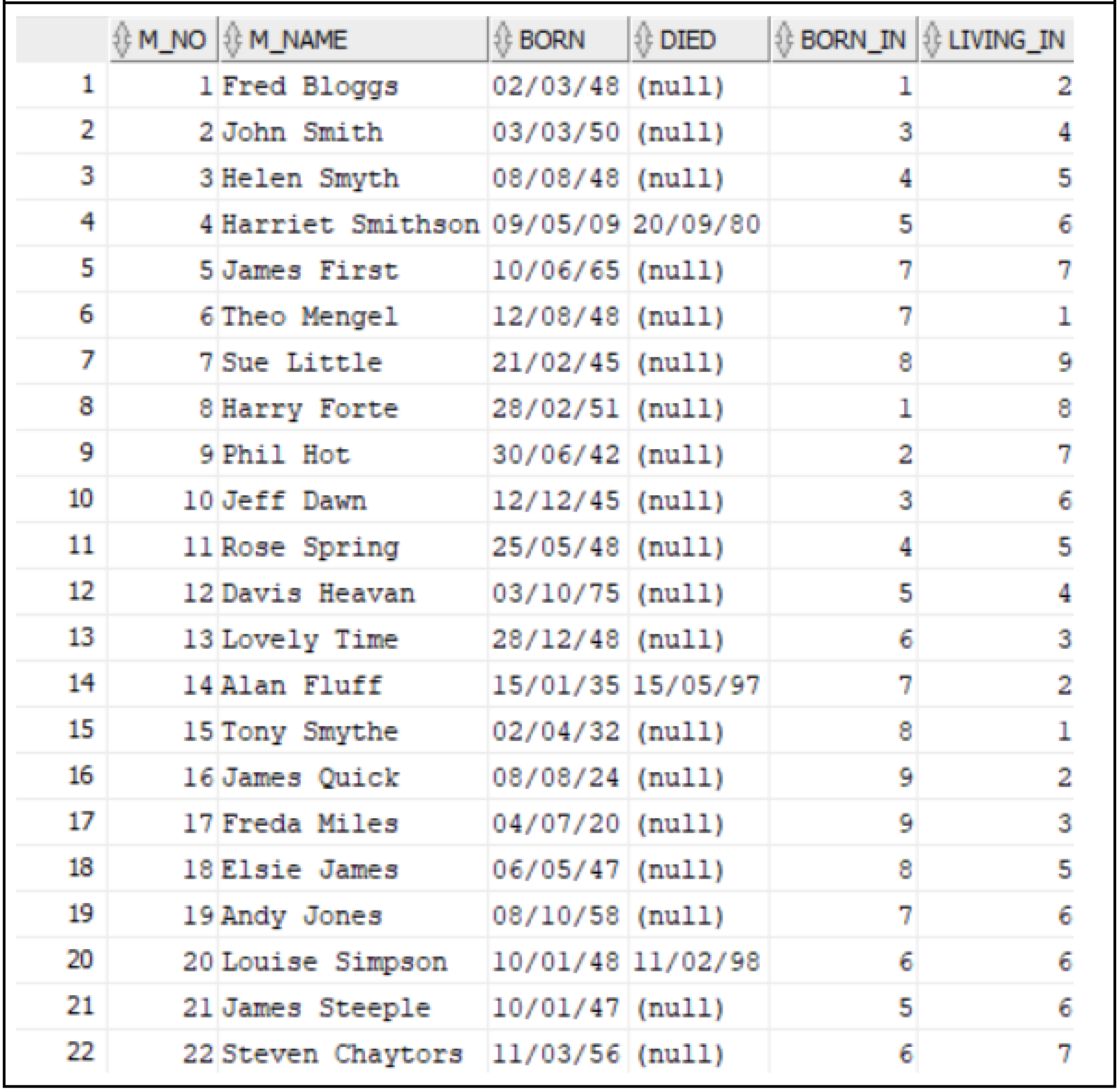
	BAND_NO	BAND_NAME	BAND_HOME	BAND_TYPE	B_DATE	BAND_CONTACT
1	1	ROP	5	classical	30/01/01	11
2	2	AASO	6	classical	(null)	10
3	3	The J Bs	8	jazz	(null)	12
4	4	BBSO	9	classical	(null)	21
5	5	The left Overs	2	jazz	(null)	8
6	6	Somebody Love this	1	jazz	(null)	6
7	7	Oh well	4	classical	(null)	3
8	8	Swinging strings	4	classical	(null)	7
9	9	The Rest	9	jazz	(null)	16



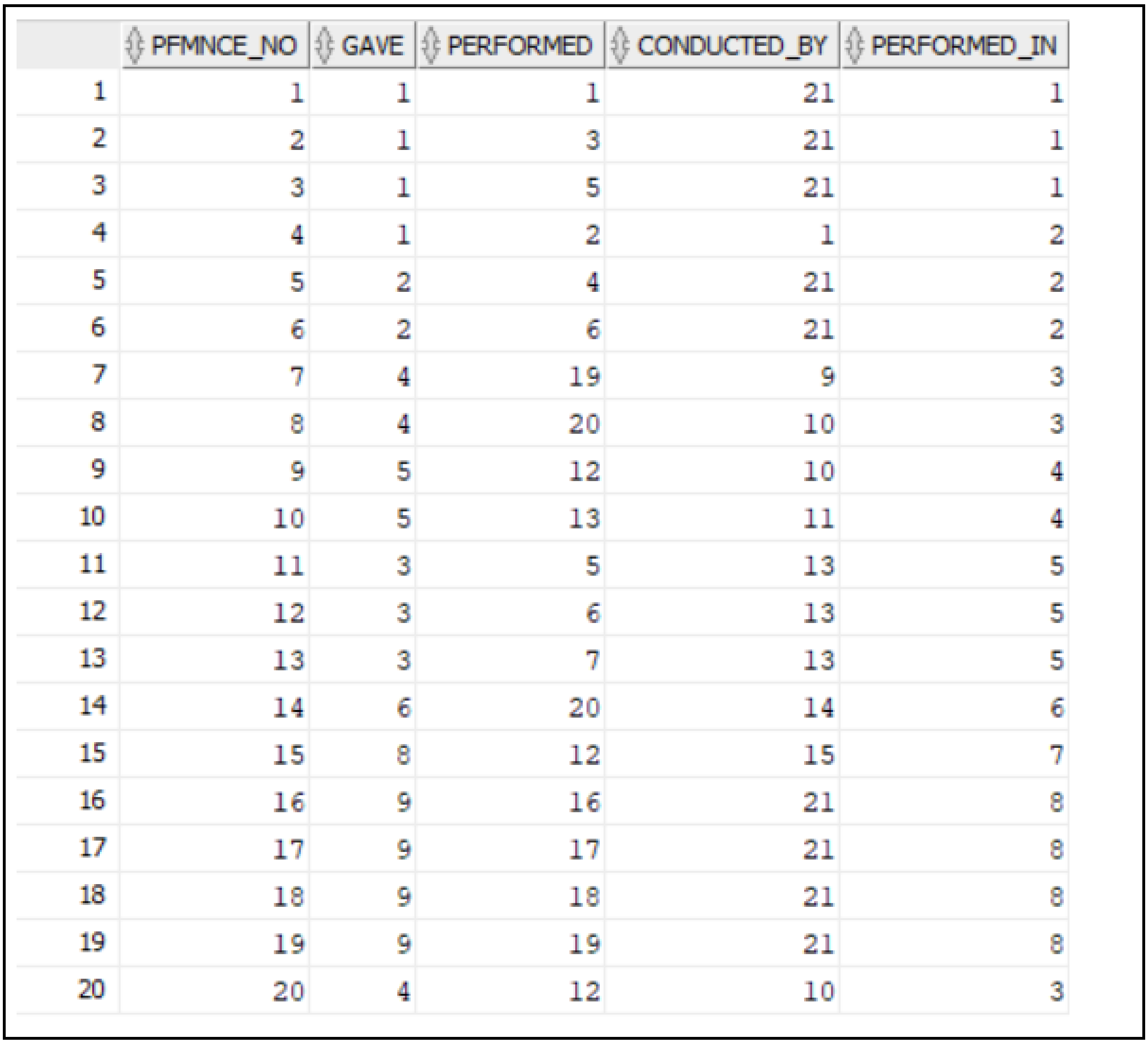
	COM...	COMP_IS	COMP_TYPE
1	1	1	jazz
2	2	3	classical
3	3	5	jazz
4	4	7	classical
5	5	9	jazz
6	6	11	rock
7	7	13	classical
8	8	15	jazz
9	9	17	classical
10	10	19	jazz
11	11	10	rock
12	12	8	jazz



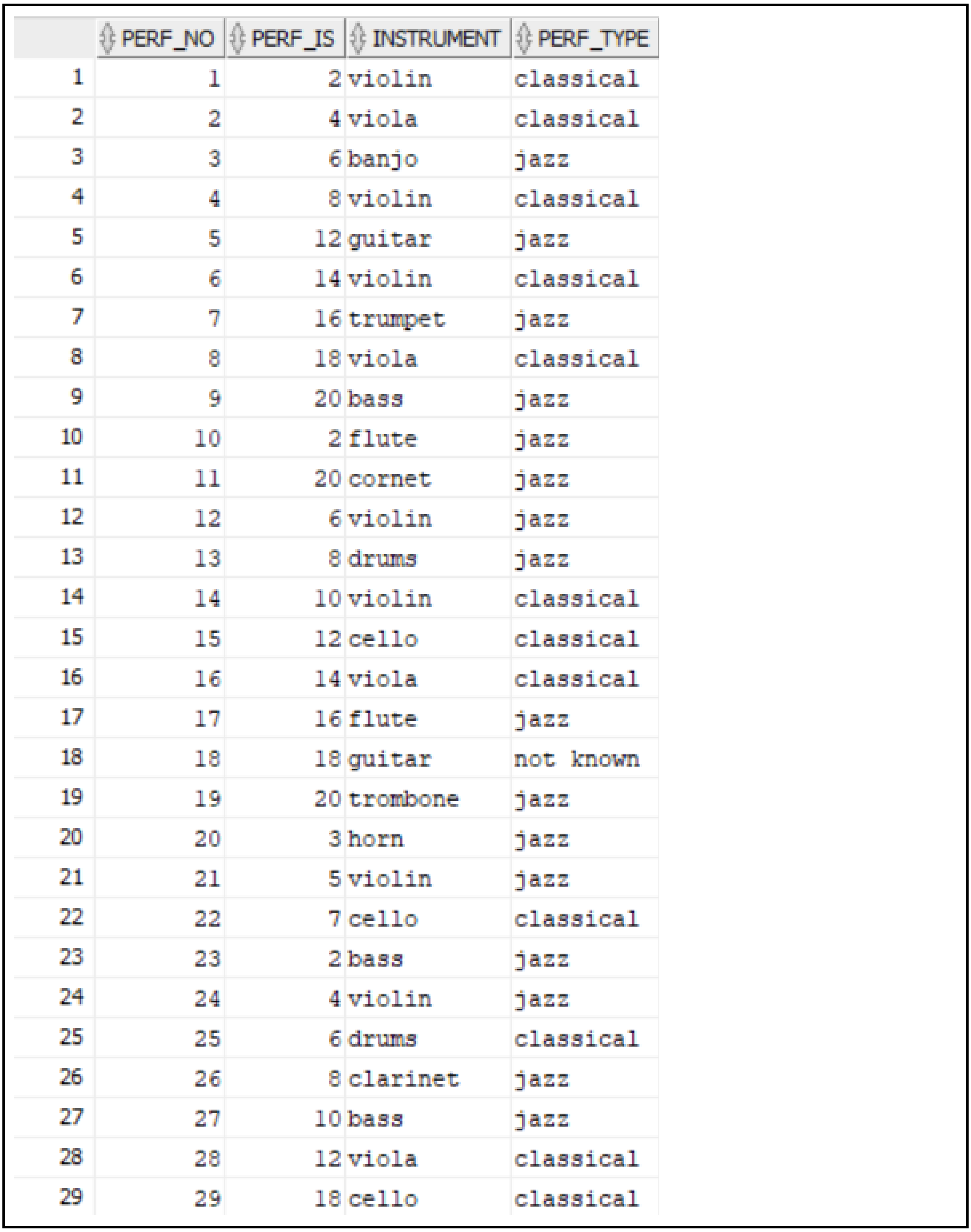
	CONCERT_NO	CONCERT_VENUE	CONCERT_IN	CON_DATE	CONCERT_ORGANISER
1	1	Bridgewater Hall	1	06/01/95	21
2	2	Bridgewater Hall	1	08/05/96	3
3	3	Usher Hall	2	03/06/95	3
4	4	Assembly Rooms	2	20/09/97	21
5	5	Festspiel Haus	3	21/02/95	8
6	6	Royal Albert Hall	7	12/04/93	8
7	7	Concertgebouw	9	14/05/93	8
8	8	Metropolitan	4	15/06/97	21



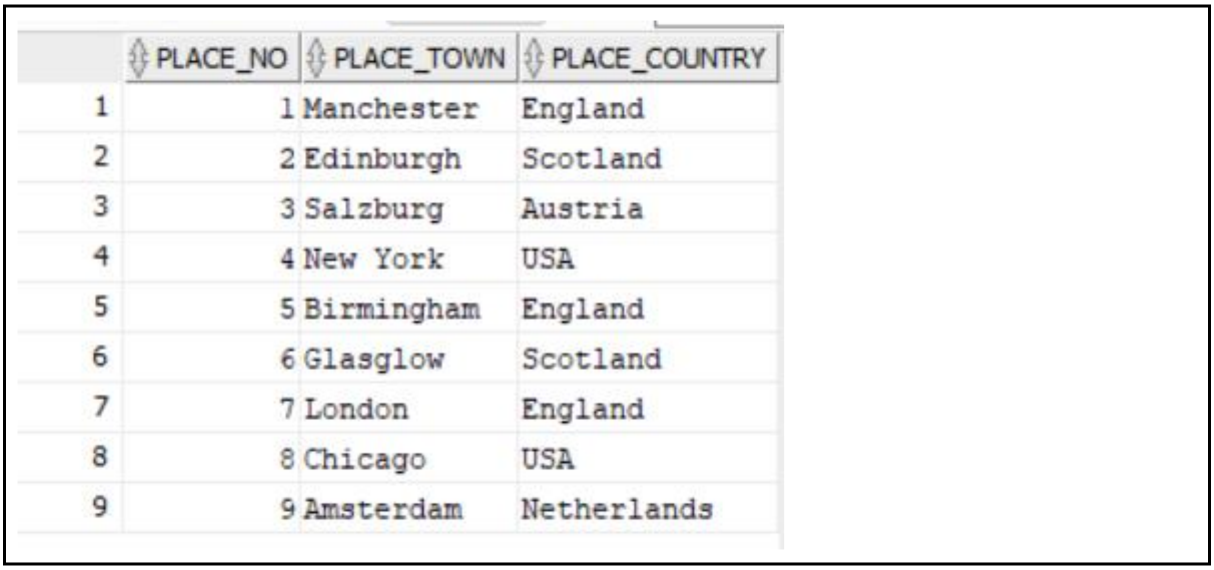
	M_NO	M_NAME	BORN	DIED	BORN_IN	LIVING_IN
1	1	Fred Bloggs	02/03/48	(null)	1	2
2	2	John Smith	03/03/50	(null)	3	4
3	3	Helen Smyth	08/08/48	(null)	4	5
4	4	Harriet Smithson	09/05/09	20/09/80	5	6
5	5	James First	10/06/65	(null)	7	7
6	6	Theo Mengel	12/08/48	(null)	7	1
7	7	Sue Little	21/02/45	(null)	8	9
8	8	Harry Forte	28/02/51	(null)	1	8
9	9	Phil Hot	30/06/42	(null)	2	7
10	10	Jeff Dawn	12/12/45	(null)	3	6
11	11	Rose Spring	25/05/48	(null)	4	5
12	12	Davis Heavan	03/10/75	(null)	5	4
13	13	Lovely Time	28/12/48	(null)	6	3
14	14	Alan Fluff	15/01/35	15/05/97	7	2
15	15	Tony Smythe	02/04/32	(null)	8	1
16	16	James Quick	08/08/24	(null)	9	2
17	17	Freda Miles	04/07/20	(null)	9	3
18	18	Elsie James	06/05/47	(null)	8	5
19	19	Andy Jones	08/10/58	(null)	7	6
20	20	Louise Simpson	10/01/48	11/02/98	6	6
21	21	James Steeple	10/01/47	(null)	5	6
22	22	Steven Chaytors	11/03/56	(null)	6	7



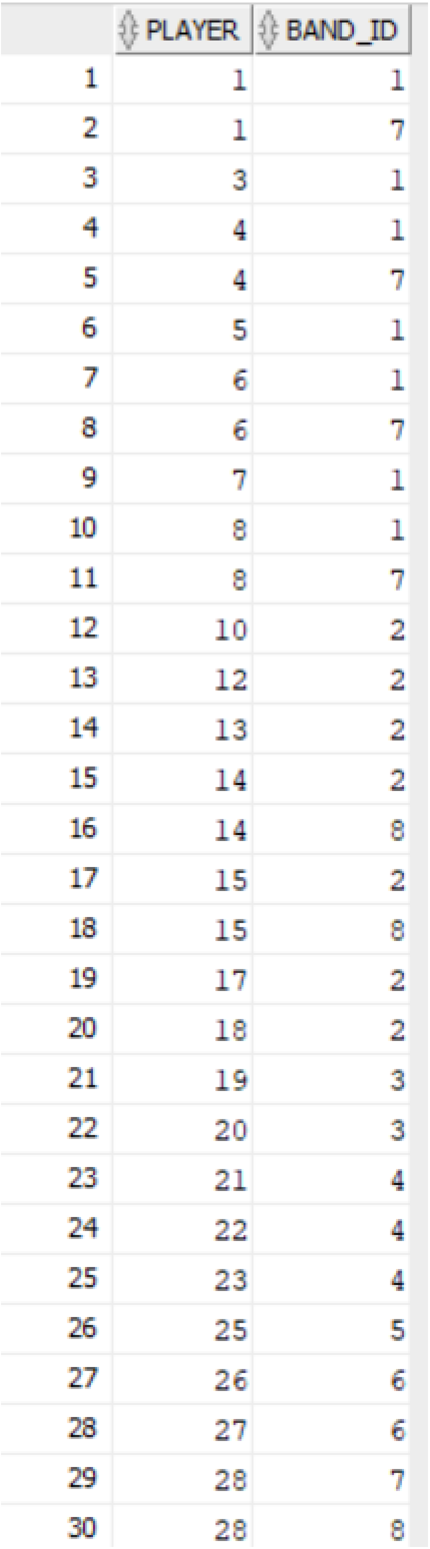
	PFMNCE_NO	GAVE	PERFORMED	CONDUCTED_BY	PERFORMED_IN
1	1	1	1	21	1
2	2	1	3	21	1
3	3	1	5	21	1
4	4	1	2	1	2
5	5	2	4	21	2
6	6	2	6	21	2
7	7	4	19	9	3
8	8	4	20	10	3
9	9	5	12	10	4
10	10	5	13	11	4
11	11	3	5	13	5
12	12	3	6	13	5
13	13	3	7	13	5
14	14	6	20	14	6
15	15	8	12	15	7
16	16	9	16	21	8
17	17	9	17	21	8
18	18	9	18	21	8
19	19	9	19	21	8
20	20	4	12	10	3



	PERF_NO	PERF_IS	INSTRUMENT	PERF_TYPE
1	1	2	violin	classical
2	2	4	viola	classical
3	3	6	banjo	jazz
4	4	8	violin	classical
5	5	12	guitar	jazz
6	6	14	violin	classical
7	7	16	trumpet	jazz
8	8	18	viola	classical
9	9	20	bass	jazz
10	10	2	flute	jazz
11	11	20	cornet	jazz
12	12	6	violin	jazz
13	13	8	drums	jazz
14	14	10	violin	classical
15	15	12	cello	classical
16	16	14	viola	classical
17	17	16	flute	jazz
18	18	18	guitar	not known
19	19	20	trombone	jazz
20	20	3	horn	jazz
21	21	5	violin	jazz
22	22	7	cello	classical
23	23	2	bass	jazz
24	24	4	violin	jazz
25	25	6	drums	classical
26	26	8	clarinet	jazz
27	27	10	bass	jazz
28	28	12	viola	classical
29	29	18	cello	classical



	PLACE_NO	PLACE_TOWN	PLACE_COUNTRY
1	1	Manchester	England
2	2	Edinburgh	Scotland
3	3	Salzburg	Austria
4	4	New York	USA
5	5	Birmingham	England
6	6	Glasglow	Scotland
7	7	London	England
8	8	Chicago	USA
9	9	Amsterdam	Netherlands

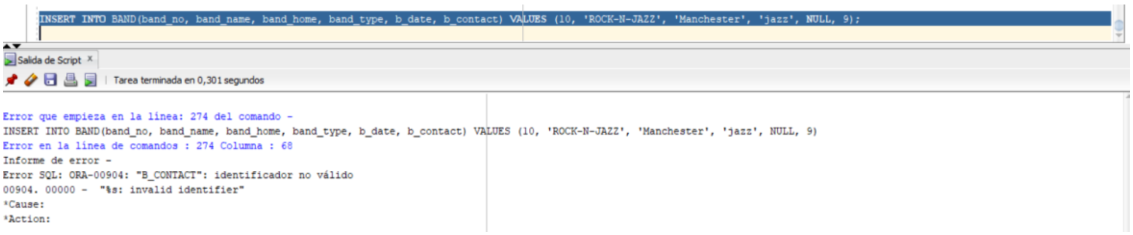


	PLAYER	BAND_ID
1	1	1
2	1	7
3	3	1
4	4	1
5	4	7
6	5	1
7	6	1
8	6	7
9	7	1
10	8	1
11	8	7
12	10	2
13	12	2
14	13	2
15	14	2
16	14	8
17	15	2
18	15	8
19	17	2
20	18	2
21	19	3
22	20	3
23	21	4
24	22	4
25	23	4
26	25	5
27	26	6
28	27	6
29	28	7
30	28	8

□ Probar algunas restricciones declarativas NoOK (PoblarNoOK)

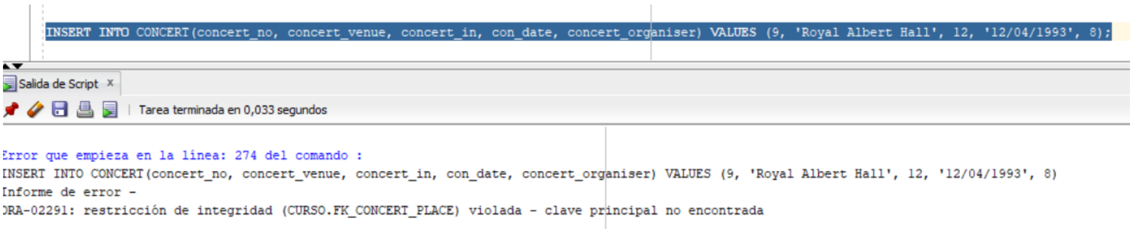
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, b_contact) VALUES (10,”ROCK-N-JAZZ”,”Manchester”,”jazz”,NULL,9);
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date,concert_orgniser) VALUES (9,”Royal Albert Hall”,12,”12/04/1993”,8);
INSERT INTO PLACE(place_no, place_town, place_country) VALUES (3,

“Bogota”,”Colombia”)
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date,concert_orgniser) VALUES (9,”Bridgewater Hall”,1,”31/02/1995”,21);
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type) VALUES (13,2,”jhcddhlklhfkvfiutgcjhñglgljvjhvvljhflhfhvlhghhglhglhyfljhvhgfljhfljhvjhvlj ljhlkjhkjhlkjlghhg hgtuyyvñfñdasjdfhglgkgñkjgkglhglhgljh”,”classical”)



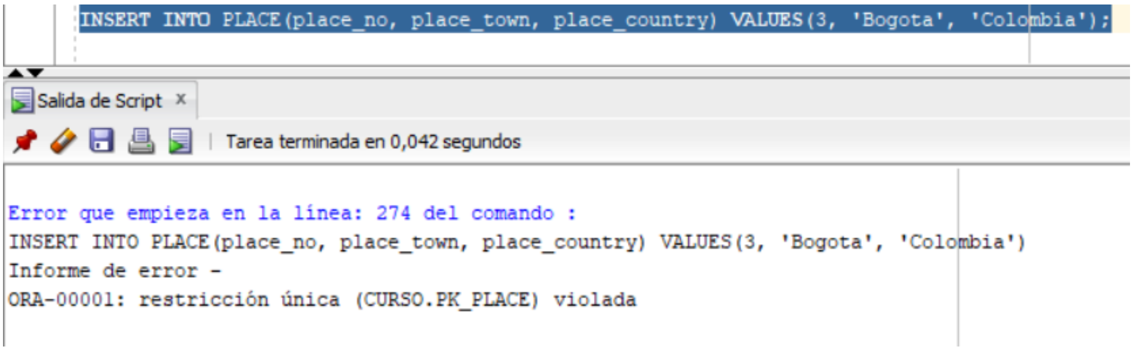
```
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, b_contact) VALUES (10, 'ROCK-N-JAZZ', 'Manchester', 'jazz', NULL, 9);
```

```
Error que empieza en la línea: 274 del comando -
INSERT INTO BAND(band_no, band_name, band_home, band_type, b_date, b_contact) VALUES (10, 'ROCK-N-JAZZ', 'Manchester', 'jazz', NULL, 9)
Error en la línea de comandos : 274 Columna : 68
Informe de error -
Error SQL: ORA-00904: "B_CONTACT": identificador no válido
00904. 00000 -  "%s: invalid identifier"
*Cause:
*Action:
```



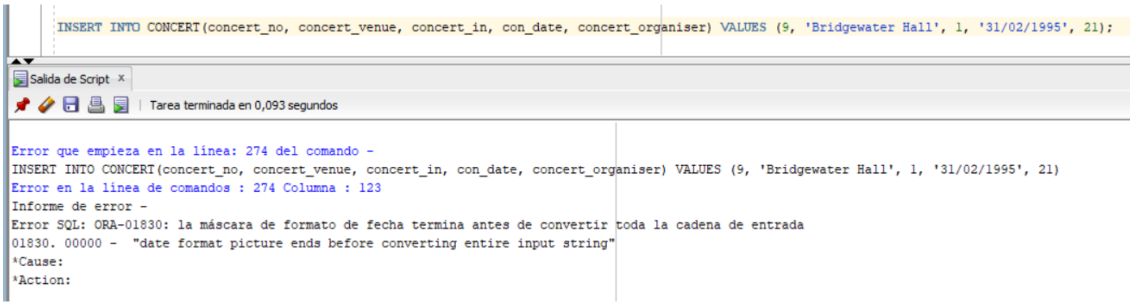
```
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date, concert_organiser) VALUES (9, 'Royal Albert Hall', 12, '12/04/1993', 8);
```

```
Error que empieza en la línea: 274 del comando :
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date, concert_organiser) VALUES (9, 'Royal Albert Hall', 12, '12/04/1993', 8)
Informe de error -
ORA-02291: restricción de integridad (CURSO.FK_CONCERT_PLACE) violada - clave principal no encontrada
```



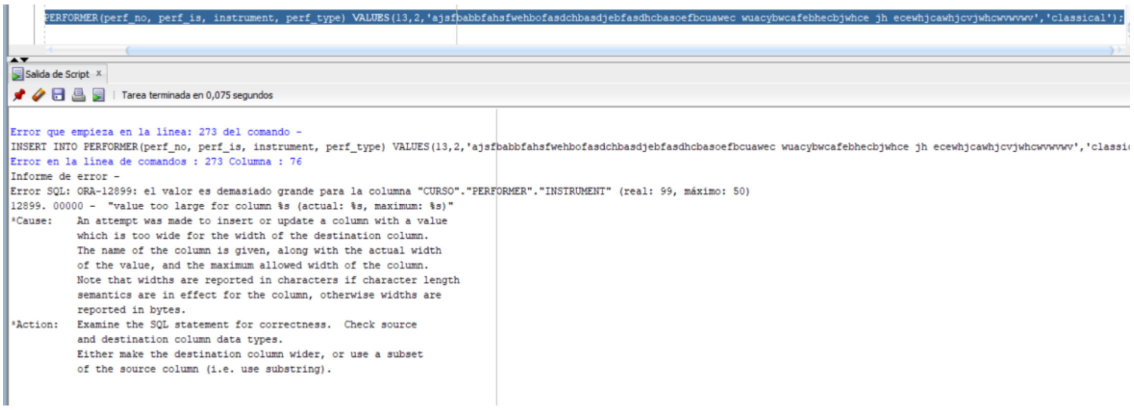
```
INSERT INTO PLACE(place_no, place_town, place_country) VALUES(3, 'Bogota', 'Colombia');
```

```
Error que empieza en la línea: 274 del comando :
INSERT INTO PLACE(place_no, place_town, place_country) VALUES(3, 'Bogota', 'Colombia')
Informe de error -
ORA-00001: restricción única (CURSO.PK_PLACE) violada
```



```
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date, concert_organiser) VALUES (9, 'Bridgewater Hall', 1, '31/02/1995', 21);
```

```
Error que empieza en la línea: 274 del comando -
INSERT INTO CONCERT(concert_no, concert_venue, concert_in, con_date, concert_organiser) VALUES (9, 'Bridgewater Hall', 1, '31/02/1995', 21)
Error en la línea de comandos : 274 Columna : 123
Informe de error -
Error SQL: ORA-01830: la máscara de formato de fecha termina antes de convertir toda la cadena de entrada
01830. 00000 -  "date format picture ends before converting entire input string"
*Cause:
*Action:
```

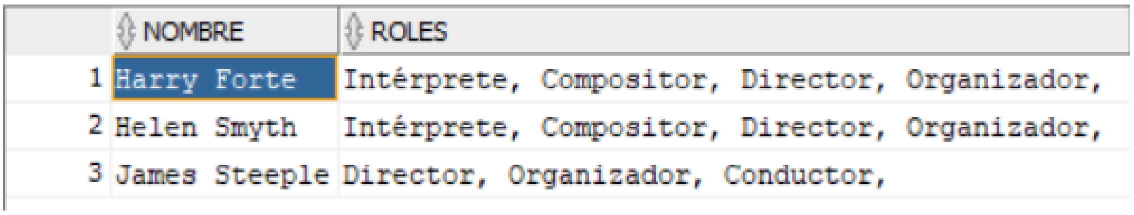


```
PERFORMER(perf_no, perf_is, instrument, perf_type) VALUES(13,2,'ajsfbabbfahsfwehbofasdchbasdjebfasdhcbasoefbcuawec wuacybwcafebhecbjwhce jh ecewhjcawhjcvjwhcwvwvwv','classical');
```

```
Error que empieza en la línea: 273 del comando -
INSERT INTO PERFORMER(perf_no, perf_is, instrument, perf_type) VALUES(13,2,'ajsfbabbfahsfwehbofasdchbasdjebfasdhcbasoefbcuawec wuacybwcafebhecbjwhce jh ecewhjcawhjcvjwhcwvwvwv','classi
Error en la línea de comandos : 273 Columna : 76
Informe de error -
Error SQL: ORA-12899: el valor es demasiado grande para la columna "CURSO"."PERFORMER"."INSTRUMENT" (real: 99, máximo: 50)
12899. 00000 -  "value too large for column %s (actual: %s, maximum: %s)"
*Cause:    An attempt was made to insert or update a column with a value
           which is too wide for the width of the destination column.
           The name of the column is given, along with the actual width
           of the value, and the maximum allowed width of the column.
           Note that widths are reported in characters if character length
           semantics are in effect for the column, otherwise widths are
           reported in bytes.
*Action:   Examine the SQL statement for correctness.  Check source
           and destination column data types.
           Either make the destination column wider, or use a subset
           of the source column (i.e. use substring).
```

□ Probar cinco consultas pertinentes para esta nueva base de datos.[1](Consultas)

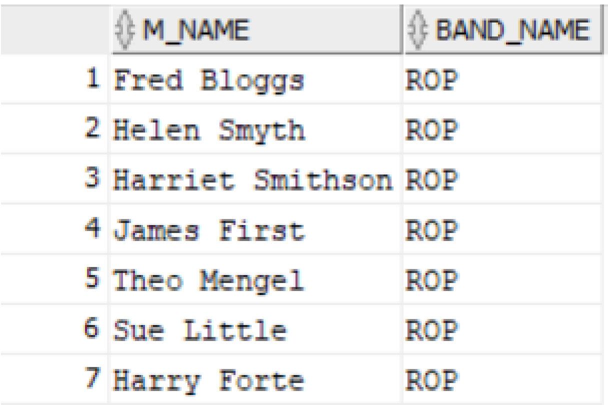
1. ¿Qué músicos han hecho el papel de organizador y director?



	NOMBRE	ROLES
1	Harry Forte	Intérprete, Compositor, Director, Organizador,
2	Helen Smyth	Intérprete, Compositor, Director, Organizador,
3	James Steeple	Director, Organizador, Conductor,

```
SELECT
    m.m_name AS Nombre,
    LTRIM(
        NVL2(p.perf_is, 'Intérprete, ', '') ||
        NVL2(c.comp_is, 'Compositor, ', '') ||
        NVL2(b.band_contact, 'Director, ', '') ||
        NVL2(co.concert_organiser, 'Organizador, ', '') ||
        NVL2(pf.conducted_by, 'Conductor, ', ''),
        ', '
    ) AS Roles
FROM
    musician m
LEFT JOIN performer p ON m.m_no = p.perf_is
LEFT JOIN composer c ON m.m_no = c.comp_is
LEFT JOIN band b ON m.m_no = b.band_contact
LEFT JOIN concert co ON m.m_no = co.concert_organiser
LEFT JOIN performance pf ON m.m_no = pf.conducted_by
WHERE (b.band_contact IS NOT NULL AND co.concert_organiser IS NOT NULL)
GROUP BY
    m.m_no,
    m.m_name,
    p.perf_is,
    c.comp_is,
    b.band_contact,
    co.concert_organiser,
    pf.conducted_by
HAVING
    LTRIM(
        NVL2(p.perf_is, 'Intérprete, ', '') ||
        NVL2(c.comp_is, 'Compositor, ', '') ||
        NVL2(b.band_contact, 'Director, ', '') ||
        NVL2(co.concert_organiser, 'Organizador, ', '') ||
        NVL2(pf.conducted_by, 'Conductor, ', ''),
        ', '
    ) IS NOT NULL
ORDER BY
    m.m_name
```

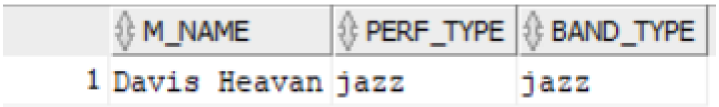
2. Consultar El nombre del músico y la banda a la que pertenece de la banda que se conoce su fecha de creación



	M_NAME	BAND_NAME
1	Fred Bloggs	ROP
2	Helen Smyth	ROP
3	Harriet Smithson	ROP
4	James First	ROP
5	Theo Mengel	ROP
6	Sue Little	ROP
7	Harry Forte	ROP

```
SELECT m_name, band_name
FROM musician
JOIN plays_in
ON m_no = player
JOIN band
ON band_id = band_no
WHERE b_date IS NOT NULL
```

3. Consultar los músicos que interpretan un género musical y coincide con el género musical de la banda a la que pertenecen

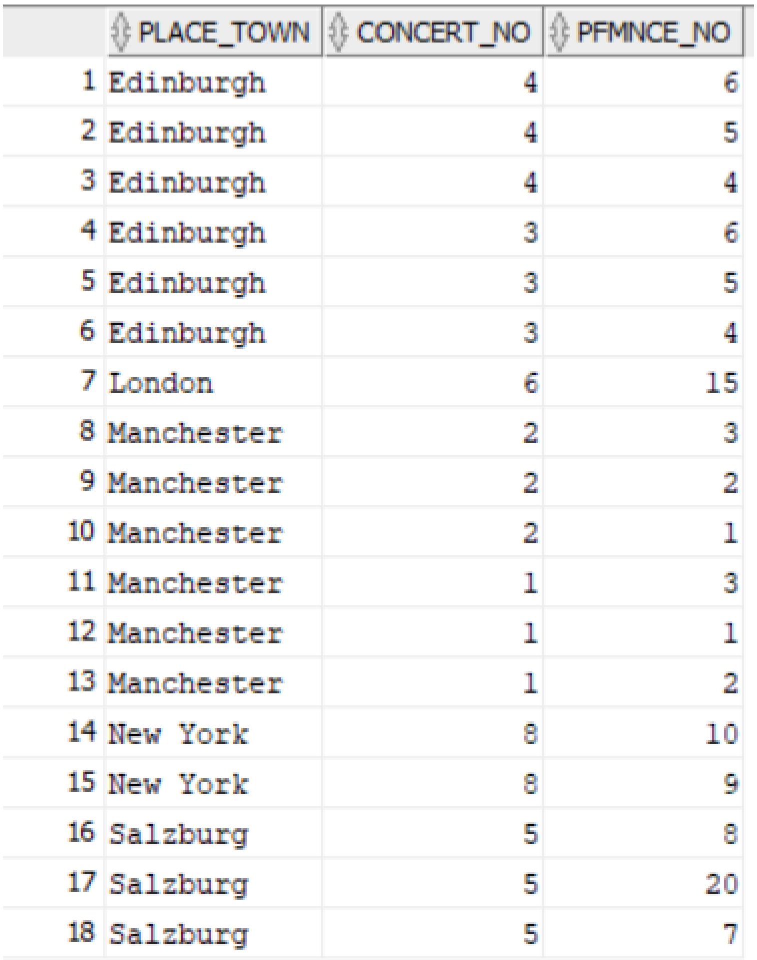


	M_NAME	PERF_TYPE	BAND_TYPE
1	Davis Heavan	jazz	jazz

```
SELECT m_name, perf_type, band_type
FROM musician
JOIN performer
ON m_no = perf_no
JOIN band
```

```
ON m_no = band_contact
WHERE perf_type = band_type
```

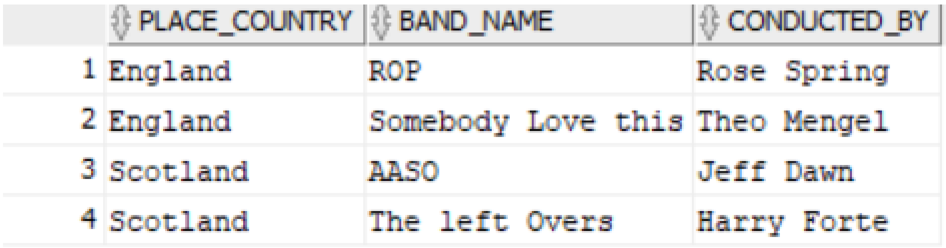
4. Consultar cuales son los conciertos e interpretaciones que se dieron en la misma ciudad.



	PLACE_TOWN	CONCERT_NO	PFMNCE_NO
1	Edinburgh	4	6
2	Edinburgh	4	5
3	Edinburgh	4	4
4	Edinburgh	3	6
5	Edinburgh	3	5
6	Edinburgh	3	4
7	London	6	15
8	Manchester	2	3
9	Manchester	2	2
10	Manchester	2	1
11	Manchester	1	3
12	Manchester	1	1
13	Manchester	1	2
14	New York	8	10
15	New York	8	9
16	Salzburg	5	8
17	Salzburg	5	20
18	Salzburg	5	7

```
SELECT DISTINCT place_town, concert_no, pfmnce_no
FROM place
JOIN concert
ON place_no = concert_in
JOIN performance
ON place_no = performed_in
GROUP BY place_town, concert_no, pfmnce_no
ORDER BY place_town
```

5. Consultar las interpretaciones (con el nombre de la banda y el músico que la condujo) que solo fueron en el Reino Unido.



	PLACE_COUNTRY	BAND_NAME	CONDUCTED_BY
1	England	ROP	Rose Spring
2	England	Somebody Love this	Theo Mengel
3	Scotland	AASO	Jeff Dawn
4	Scotland	The left Overs	Harry Forte

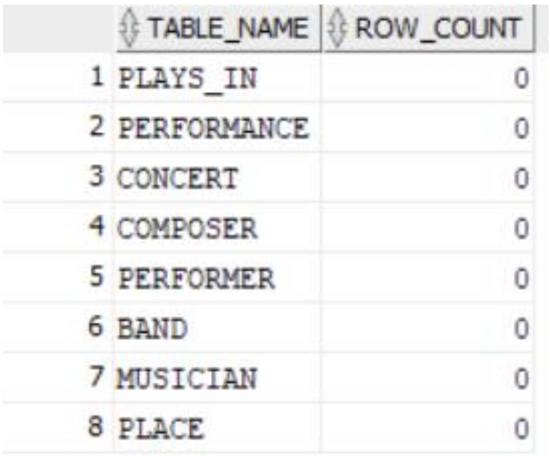
```
SELECT place_country, band_name, m_name AS Conducted_by
FROM place
JOIN band
ON place_no = band_home
JOIN musician
ON band_contact = m_no
WHERE place_country IN ('England', 'Scotland')
ORDER BY place_country, band_name, Conducted_by
```

□ Despoblar la base de datos (XPoblar)

```
ALTER TABLE PLAYS_IN DISABLE CONSTRAINT
FK_PERFORMER_PLAY;
ALTER TABLE PLAYS_IN DISABLE CONSTRAINT
FK_BAND_PLAY;
ALTER TABLE MUSICIAN DISABLE CONSTRAINT
FK1_MUSICIAN_PLACE;
ALTER TABLE MUSICIAN DISABLE CONSTRAINT
FK2_MUSICIAN_PLACE;
ALTER TABLE CONCERT DISABLE CONSTRAINT
FK_CONCERT_PLACE;
ALTER TABLE PERFORMANCE DISABLE CONSTRAINT
FK_PERFORMANCE_PLACE;
ALTER TABLE BAND DISABLE CONSTRAINT FK_BAND_PLACE;
ALTER TABLE PERFORMANCE DISABLE CONSTRAINT
FK_PERFORMANCE_BAND;
ALTER TABLE PERFORMANCE DISABLE CONSTRAINT
FK_PERFORMANCE_MUSICIAN;
ALTER TABLE COMPOSER DISABLE CONSTRAINT
FK_COMPOSER_MUSICIAN;
ALTER TABLE PERFORMER DISABLE CONSTRAINT
FK_PERFORMER_MUSICIAN;
ALTER TABLE CONCERT DISABLE CONSTRAINT
FK_CONCERT_MUSICIAN;
ALTER TABLE BAND DISABLE CONSTRAINT
FK_BAND_MUSICIAN;


TRUNCATE TABLE PLAYS_IN;
TRUNCATE TABLE PERFORMANCE;
TRUNCATE TABLE CONCERT;
TRUNCATE TABLE COMPOSER;
TRUNCATE TABLE PERFORMER;
TRUNCATE TABLE BAND;
TRUNCATE TABLE MUSICIAN;
TRUNCATE TABLE PLACE;
```

```
SELECT 'PLAYS_IN' AS table_name, COUNT(*) AS row_count FROM
PLAYS_IN
UNION ALL
SELECT 'PERFORMANCE', COUNT(*) FROM PERFORMANCE
UNION ALL
SELECT 'CONCERT', COUNT(*) FROM CONCERT
UNION ALL
SELECT 'COMPOSER', COUNT(*) FROM COMPOSER
UNION ALL
SELECT 'PERFORMER', COUNT(*) FROM PERFORMER
UNION ALL
SELECT 'BAND', COUNT(*) FROM BAND
UNION ALL
SELECT 'MUSICIAN', COUNT(*) FROM MUSICIAN
UNION ALL
SELECT 'PLACE', COUNT(*) FROM PLACE;
```

Fui eliminando parte por parte la información de la base de datos. Primero las restricciones entre tablas (toda la parte de ALTER TABLE **** DISABLE CONSTRAINT), luego eliminé toda la información de las tablas (TRUNCATE TABLE) y para comprobar que no quedaron filas asociadas, usé el comando de COUNT(*) en cada una de las tablas, lo cual me dió el siguiente resultado:

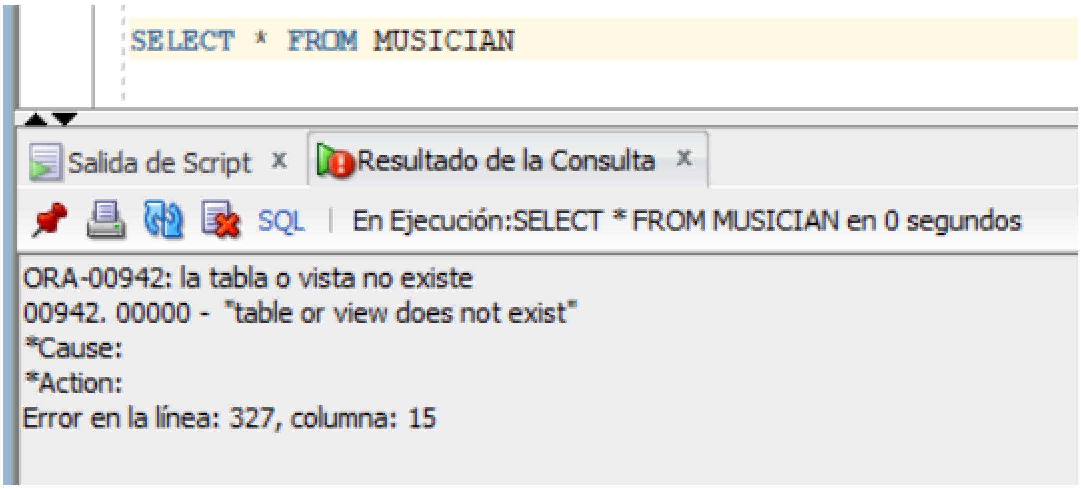


	TABLE_NAME	ROW_COUNT
1	PLAYS_IN	0
2	PERFORMANCE	0
3	CONCERT	0
4	COMPOSER	0
5	PERFORMER	0
6	BAND	0
7	MUSICIAN	0
8	PLACE	0

- Eliminar toda la información de la base de datos (XTablas)

Por último, para eliminar todas las tablas de la base de datos se coloca DROP TABLE **** y se eliminan todas las tablas agregadas.

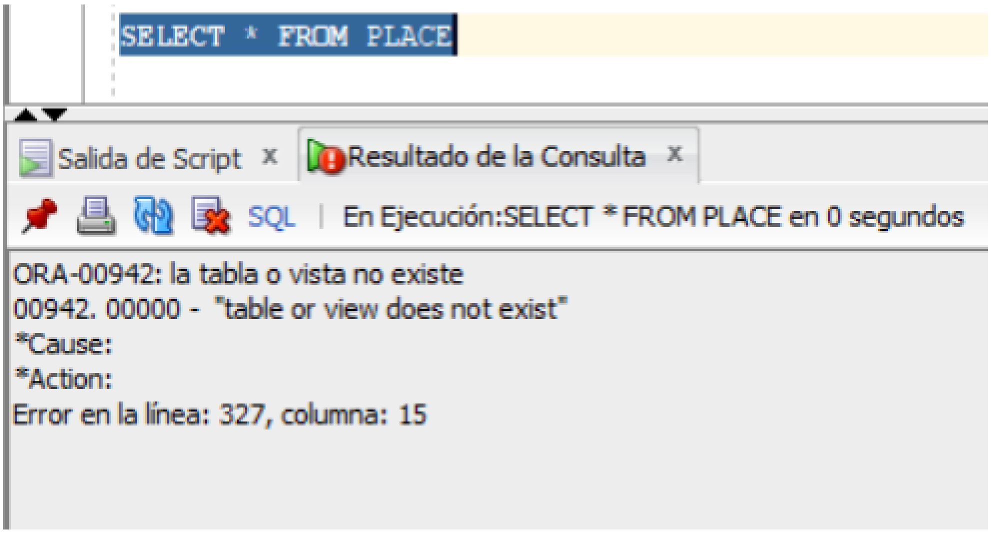
```
DROP TABLE PLAYS_IN;
DROP TABLE PERFORMANCE;
DROP TABLE PERFORMER;
DROP TABLE MUSICIAN;
DROP TABLE COMPOSER;
DROP TABLE CONCERT;
DROP TABLE PLACE;
DROP TABLE BAND;
```



```
SELECT * FROM MUSICIAN
```

En Ejecución:SELECT * FROM MUSICIAN en 0 segundos

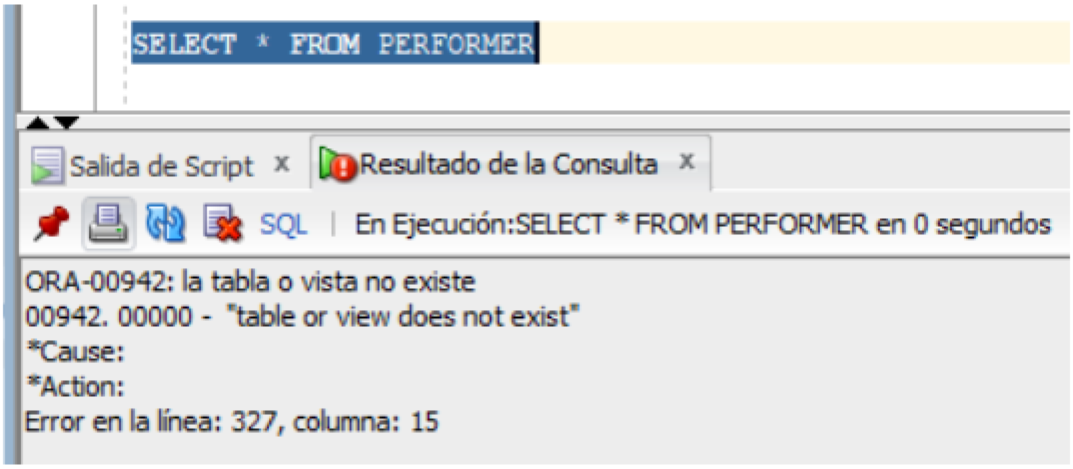
```
ORA-00942: la tabla o vista no existe
00942. 00000 -  "table or view does not exist"
*Cause:
*Action:
Error en la línea: 327, columna: 15
```



```
SELECT * FROM PLACE
```

En Ejecución:SELECT * FROM PLACE en 0 segundos

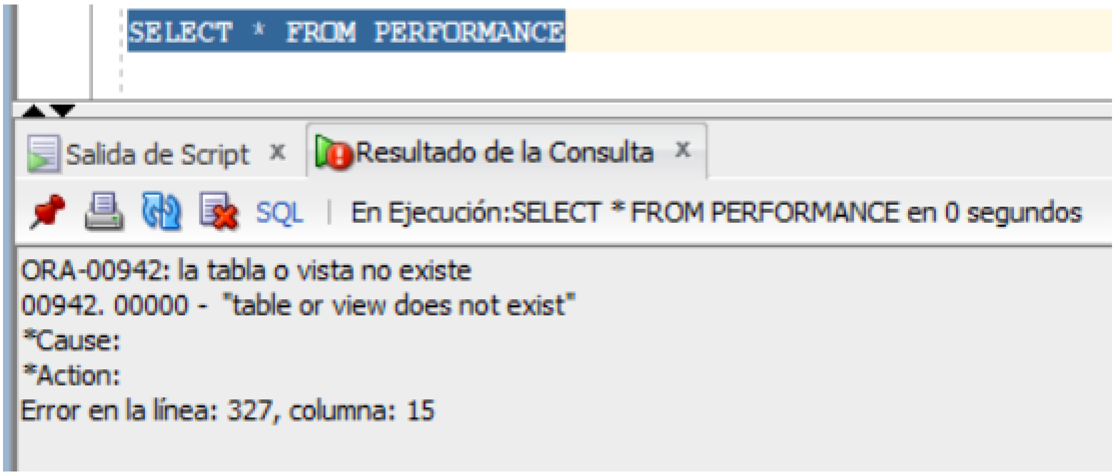
```
ORA-00942: la tabla o vista no existe
00942. 00000 -  "table or view does not exist"
*Cause:
*Action:
Error en la línea: 327, columna: 15
```



```
SELECT * FROM PERFORMER
```

En Ejecución:SELECT * FROM PERFORMER en 0 segundos

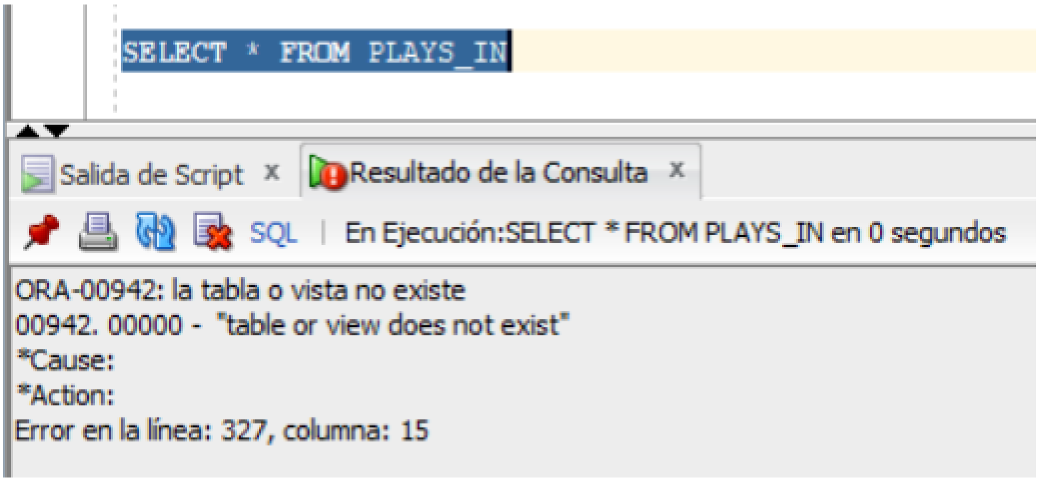
```
ORA-00942: la tabla o vista no existe
00942. 00000 -  "table or view does not exist"
*Cause:
*Action:
Error en la línea: 327, columna: 15
```



```
SELECT * FROM PERFORMANCE
```

En Ejecución:SELECT * FROM PERFORMANCE en 0 segundos

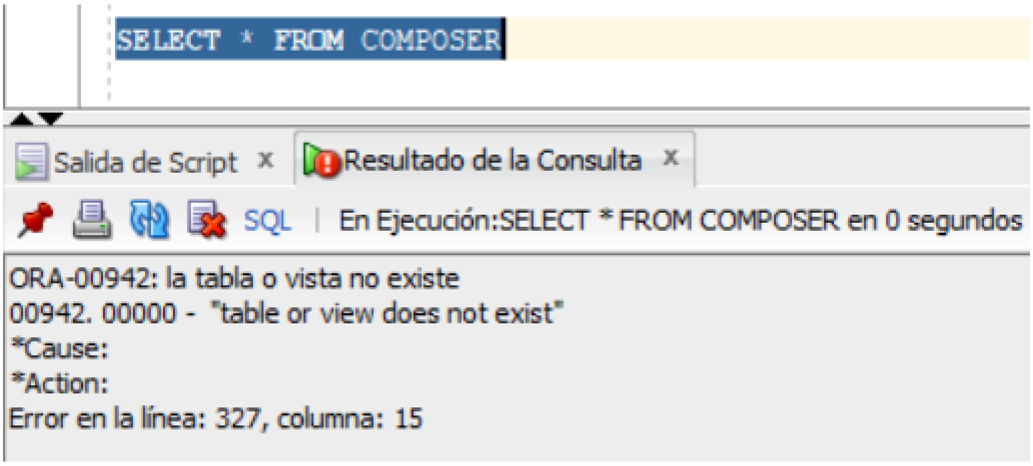
```
ORA-00942: la tabla o vista no existe
00942. 00000 -  "table or view does not exist"
*Cause:
*Action:
Error en la línea: 327, columna: 15
```



```
SELECT * FROM PLAYS_IN
```

En Ejecución:SELECT * FROM PLAYS_IN en 0 segundos

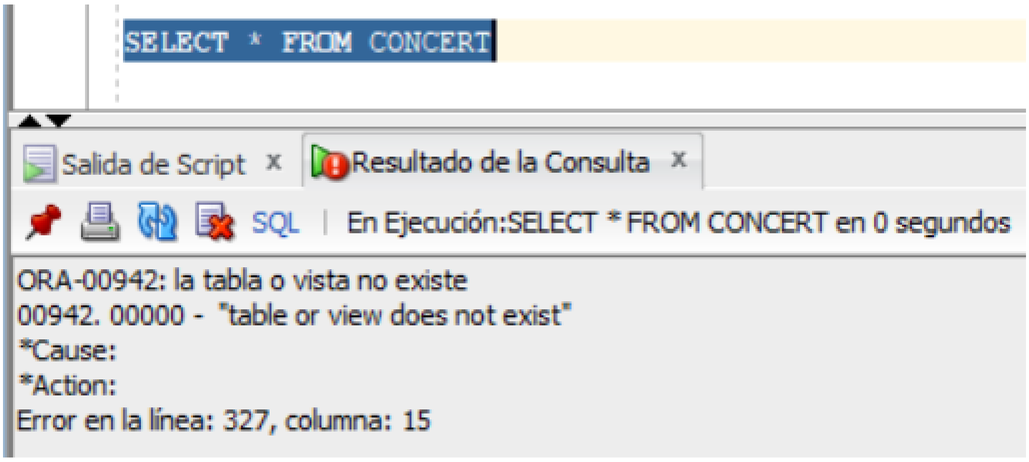
```
ORA-00942: la tabla o vista no existe
00942. 00000 -  "table or view does not exist"
*Cause:
*Action:
Error en la línea: 327, columna: 15
```



```
SELECT * FROM COMPOSER
```

En Ejecución:SELECT * FROM COMPOSER en 0 segundos

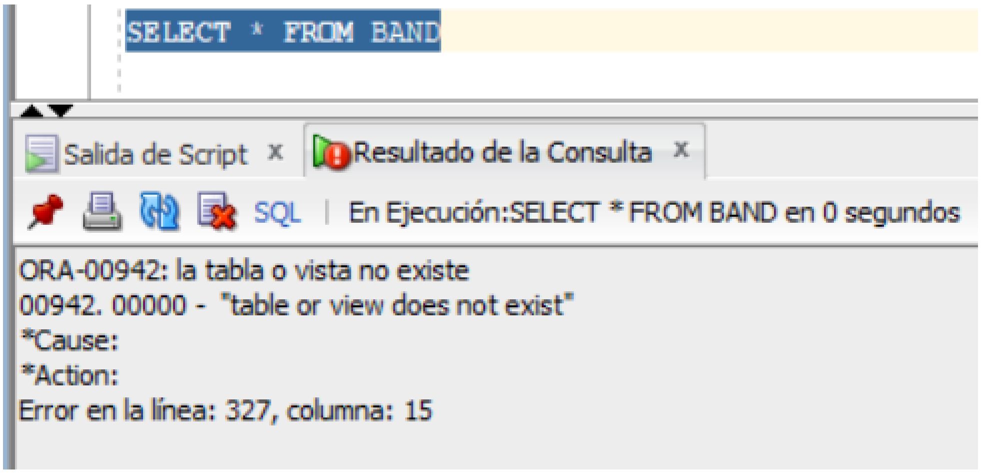
```
ORA-00942: la tabla o vista no existe
00942. 00000 -  "table or view does not exist"
*Cause:
*Action:
Error en la línea: 327, columna: 15
```



```
SELECT * FROM CONCERT
```

En Ejecución:SELECT * FROM CONCERT en 0 segundos

```
ORA-00942: la tabla o vista no existe
00942. 00000 -  "table or view does not exist"
*Cause:
*Action:
Error en la línea: 327, columna: 15
```



```
SELECT * FROM BAND
```

En Ejecución:SELECT * FROM BAND en 0 segundos

```
ORA-00942: la tabla o vista no existe
00942. 00000 -  "table or view does not exist"
*Cause:
*Action:
Error en la línea: 327, columna: 15
```