UNIVERSIDAD DEL PACÍFICO
FACULTAD DE INGENIERÍA

Ingeniería de Datos

Facultad de Ingeniería - 2019 I

Ingresar datos en SQL Server 2014

1. Descripción de la base de datos

En este trabajo práctico utilizaremos la base de datos que construimos en la práctica anterior (Viajes Luca). Recordemos que nuestro sistema permitirá la adquisición de pasajes aéreos a fin de conocer las preferencias de nuestros clientes (pasajeros). El modelo relacional que utilizaremos para construir nuestra base de datos es la siguiente:

- COMPANY (ID_comp, name)
- PASSENGER (ID_psg, name)
- PASS_IN_TRIP (trip_no, date, ID_psg, place)
- TRIP (trip_no, ID_comp, plane, town_from, town_to, time_out, time_in)

Una vez la base de datos construida, debemos insertar datos dentro de cada tabla. Para ello, utilizamos el comando *INSERT INTO*. La sintaxis del comando a sido descrita en el curso teórico.

NOTA: Si decide copiar/pegar las sentencias, puede tener problemas con las comillas (UTF-8) las cuales deben ser reemplazadas por comillas simples.

Primero agregamos datos a nuestra tabla *Company*.

```
insert into Company values(1,'Don_avia');
insert into Company values(2,'Aeroflot');
insert into Company values(3,'Dale_avia');
insert into Company values(4,'air_France');
insert into Company values(5,'British_AW');
```

Luego, agregamos datos de nuestra lujosa cartera de clientes.

```
insert into Passenger values(1,'Bruce Willis');
insert into Passenger values(2,'George Clooney');
insert into Passenger values(3,'Kevin Costner');
insert into Passenger values(4,'Donald Sutherland');
insert into Passenger values(5,'Jennifer Lopez');
insert into Passenger values(6,'Ray Liotta ');
insert into Passenger values(7,'Samuel L. Jackson');
insert into Passenger values(8,'Nikole Kidman');
insert into Passenger values(9,'Alan Rickman');
insert into Passenger values(10,'Kurt Russell');
insert into Passenger values(11,'Harrison Ford');
insert into Passenger values(12,'Russell Crowe');
insert into Passenger values(13,'Steve Martin');
insert into Passenger values(14,'Michael Caine');
```

```
insert into Passenger values(15,'Angelina Jolie');
insert into Passenger values(16,'Mel Gibson');
insert into Passenger values(17,'Michael Douglas');
insert into Passenger values(18,'John Travolta');
insert into Passenger values(19,'Sylvester Stallone');
insert into Passenger values(20,'Tommy Lee Jones');
insert into Passenger values(21,'Catherine Zeta-Jones');
insert into Passenger values(22,'Antonio Banderas');
insert into Passenger values(23,'Kim Basinger');
insert into Passenger values(24,'Sam Neill');
insert into Passenger values(25,'Gary Oldman');
insert into Passenger values(26,'Clint Eastwood');
insert into Passenger values(27,'Brad Pitt');
insert into Passenger values(28,'Johnny Depp');
insert into Passenger values(29,'Pierce Brosnan');
insert into Passenger values(30,'Sean Connery');
insert into Passenger values(31,'Bruce Willis');
insert into Passenger values(37,'Mullah Omar');
```

Ahora, la lista de vuelos.

```
insert into Trip values(1100,4,'Boeing','Rostov','Paris','19000101 14:30:00.000','19000101 17:50:00.000');
insert into Trip values(1101,4,'Boeing','Paris','Rostov','19000101 08:12:00.000','19000101 11:45:00.000');
insert into Trip values(1123,3,'TU-154','Rostov','Vladivostok','19000101 16:20:00.000','19000101 03:40:00.000');
insert into Trip values(1124,3,'TU-154','Vladivostok','Rostov','19000101 09:00:00.000','19000101 19:50:00.000');
insert into Trip values(1145,2,'IL-86','Moscow','Rostov','19000101 09:35:00.000','19000101 11:23:00.000');
insert into Trip values(1146,2,'IL-86','Rostov','Moscow','19000101 17:55:00.000','19000101 20:01:00.000');
insert into Trip values(1181,1,'TU-134','Rostov','Moscow','19000101 06:12:00.000','19000101 08:01:00.000');
insert into Trip values(1182,1,'TU-134','Moscow','Rostov','19000101 12:35:00.000','19000101 14:30:00.000');
insert into Trip values(1187,1,'TU-134','Rostov','Moscow','19000101 15:42:00.000','19000101 17:39:00.000');
insert into Trip values(1188,1,'TU-134','Moscow','Rostov','19000101 22:50:00.000','19000101 00:48:00.000');
insert into Trip values(1195,1,'TU-154','Rostov','Moscow','19000101 23:30:00.000','19000101 01:11:00.000');
insert into Trip values(1196,1,'TU-154','Moscow','Rostov','19000101 04:00:00.000','19000101 05:45:00.000');
insert into Trip values(7771,5,'Boeing','London','Singapore','19000101 01:00:00.000','19000101 11:00:00.000');
insert into Trip values(7772,5,'Boeing','Singapore','London','19000101 12:00:00.000','19000101 02:00:00.000');
insert into Trip values(7773,5,'Boeing','London','Singapore','19000101 03:00:00.000','19000101 13:00:00.000');
insert into Trip values(7774,5,'Boeing','Singapore','London','19000101 14:00:00.000','19000101 06:00:00.000');
insert into Trip values(7775,5,'Boeing','London','Singapore','19000101 09:00:00.000','19000101 20:00:00.000');
insert into Trip values(7776,5,'Boeing','Singapore','London','19000101 18:00:00.000','19000101 08:00:00.000');
insert into Trip values(7777,5,'Boeing','London','Singapore ','19000101 18:00:00.000','19000101 06:00:00.000');
insert into Trip values(7778,5,'Boeing','Singapore','London','19000101 22:00:00.000','19000101 12:00:00.000');
insert into Trip values(8881,5,'Boeing','London','Paris','19000101 03:00:00.000','19000101 04:00:00.000');
insert into Trip values(8882,5,'Boeing','Paris','London','19000101 22:00:00.000','19000101 23:00:00.000');
```

Finalmente, agregamos los pasajes vendidos a nuestros clientes.

```
insert into Pass_in_trip values(1100,'20030429 00:00:00.000',1,'1a');
insert into Pass_in_trip values(1123,'20030405 00:00:00.000',3,'2a');
insert into Pass_in_trip values(1123,'20030408 00:00:00.000',1,'4c');
insert into Pass_in_trip values(1123,'20030408 00:00:00.000',6,'4b');
insert into Pass_in_trip values(1124,'20030402 00:00:00.000',2,'2d');
insert into Pass_in_trip values(1145,'20030405 00:00:00.000',3,'2c');
insert into Pass_in_trip values(1181,'20030401 00:00:00.000',1,'1a');
insert into Pass_in_trip values(1181,'20030401 00:00:00.000',6,'1b');
insert into Pass_in_trip values(1181,'20030401 00:00:00.000',8,'3c');
insert into Pass_in_trip values(1181,'20030413 00:00:00.000',5,'1b');
insert into Pass_in_trip values(1182,'20030413 00:00:00.000',5,'4b');
```

```
insert into Pass_in_trip values(1187,'20030414 00:00:00.000',8,'3a');
insert into Pass_in_trip values(1188,'20030401 00:00:00.000',8,'3a');
insert into Pass_in_trip values(1182,'20030413 00:00:00.000',9,'6d');
insert into Pass_in_trip values(1145,'20030425 00:00:00.000',5,'1d');
insert into Pass_in_trip values(1187,'20030414 00:00:00.000',10,'3d');
insert into Pass_in_trip values(8882,'20051106 00:00:00.000',37,'1a') ;
insert into Pass_in_trip values(7771,'20051107 00:00:00.000',37,'1c') ;
insert into Pass_in_trip values(7772,'20051107 00:00:00.000',37,'1a') ;
insert into Pass_in_trip values(8881,'20051108 00:00:00.000',37,'1d') ;
insert into Pass_in_trip values(7778,'20051105 00:00:00.000',10,'2a') ;
insert into Pass_in_trip values(7772,'20051129 00:00:00.000',10,'3a');
insert into Pass_in_trip values(7771,'20051104 00:00:00.000',11,'4a');
insert into Pass_in_trip values(7771,'20051107 00:00:00.000',11,'1b');
insert into Pass_in_trip values(7771,'20051109 00:00:00.000',11,'5a');
insert into Pass_in_trip values(7772,'20051107 00:00:00.000',12,'1d');
insert into Pass_in_trip values(7773,'20051107 00:00:00.000',13,'2d');
insert into Pass_in_trip values(7772,'20051129 00:00:00.000',13,'1b');
insert into Pass_in_trip values(8882,'20051113 00:00:00.000',14,'3d');
insert into Pass_in_trip values(7771,'20051114 00:00:00.000',14,'4d');
insert into Pass_in_trip values(7771,'20051116 00:00:00.000',14,'5d');
insert into Pass_in_trip values(7772,'20051129 00:00:00.000',14,'1c');
```

NOTA: Seguramente notaron que las tablas *Trip* y *Pass_in_trip* tienen datos que son erróneos. Por ejemplo, las horas de la columna *Pass_in_trip.date* tienen un valor de *00:00:00.000*. Esto debido a que esta tabla solo debe almacenar la fecha, mas no la hora de salida y llegada de un vuelo, la cual se encuentra en la tabla *Trip*. Este error se cometió expresamente porque será corregido en otro trabajo práctico.

2. Consultas simples en SQL

Una vez creada la base de datos, las tablas, las relaciones y con datos en las tablas, ahora debemos realizar consultas a fin de obtener información que responda ciertas preguntas. Para ello, continuamos sobre la misma ventana de edición de consultas SQL.

La primera consulta que haremos (y la más sencilla) permitirá mostrar todos los atributos de todos nuestros clientes (pasajeros). La sentencia SQL que nos muestra el resultado es:

```
select * from Passenger ;
```

El resultado puede visualizarse en la ventana de resultados de *SQL Server 2014*.

Ahora, queremos saber cuáles son los viajes (trips) cuya ciudad de salida sea Moscú. La sentencia siguiente nos brinda esta información.

```
select * from Trip where town_from = 'Moscow';
```

Y qué escribiríamos si deseamos tener la información de los vuelos que salen de Moscú y cuyo avión sea un IL-86?

```
select * from Trip where town_from = 'Moscow' and plane = 'IL-86';
```

Por otro lado, deseamos conocer el código del pasajero cuyo nombre es Bruce Willis. ¿Qué escribiríamos?

```
select ID_psg from Passenger where name = 'Bruce Willis';
```

Ejercicios

1. Mostrar el nombre de los pasajeros cuyo código sea mayor a 15.
2. Mostrar el nombre de los pasajeros cuyo código esté en el rango de 15 a 30.
3. Mostrar el código del pasajero Kurt Russell.
4. Mostrar las ciudades de llegada de los vuelos que parten de Londres (London).
5. Mostrar las ciudades de partida de los vuelos que llegan a Londres (London).
6. Mostrar las tipo de avión de que son utilizados en los vuelos que parten de Paris.
7. Mostrar las tipo de avión de que son utilizados en los vuelos que llegan a Paris.
8. Mostrar el código de la compañía y el tipo de avión de los viajes realizados entre Moscow y Rostov.
9. Mostrar el código de los pasajeros que ocuparon la silla 3a
10. Mostrar el nombre de los pasajeros que empiecen con la letra "M".

3. Consultas usando varias tablas en SQL

Hasta el momento, hemos buscado información de una sola tabla. Pero, cómo podemos juntar dos tablas y mezclar información de ambas? Podemos utilizar las relaciones creadas mediante el uso de los *primary key* y los *foreign key* junto con la operación *JOIN* de SQL.

Por ejemplo, si deseamos mostrar "todos" los datos de los vuelos y las compañías aéreas, podemos escribir la siguiente sentencia SQL.

```
select * from Company join Trip on Company.ID_comp = Trip.ID_comp;
```

La consulta anterior nos muestra todos los atributos de la operación *JOIN* sobre las tablas *Company* y *Trip*, incluyendo el atributo que permite hacer dicha operación (por duplicado). En la sentencia anterior, también podemos notar que es necesaria la utilización de la cláusula *ON* para hacer referencia a los atributos que permiten hacer el *JOIN*.

Para seleccionar solo algunas columnas, podemos utilizar la proyección (select). Por ejemplo:

```
select Company.name, Trip.plane, Trip.town_from, Trip.town_to
from Company join Trip on Company.ID_comp = Trip.ID_comp;
```

En la consulta anterior, queremos mostrar solamente los viajes que se realizaron en un avión *Boeing* o *TU-134*. Entonces, debemos agregar la cláusula *WHERE* y escribimos lo siguiente:

```
select Company.name, Trip.plane, Trip.town_from, Trip.town_to
from Company join Trip on Company.ID_comp = Trip.ID_comp
where Trip.plane = 'Boeing' or Trip.plane = 'TU-134';
```

En el resultado anterior, tenemos justamente lo que hemos solicitado. Imaginemos que, ahora queremos ordenar los datos en forma ascendente o descendente. La cláusula *ORDER BY* permite hacer eso. Esta cláusula tiene dos opciones: *ASC* y *DESC*, las cuales ordenan los resultados de manera ascendente y descendente respectivamente. Por ejemplo, si queremos ordenar los resultados de nuestra consulta anterior, en función del tipo de avión y de manera ascendente, podemos utilizar la siguiente sentencia.

```
select Company.name, Trip.plane, Trip.town_from, Trip.town_to
from Company join Trip on Company.ID_comp = Trip.ID_comp
where Trip.plane = 'Boeing' or Trip.plane = 'TU-134'
order by Trip.plane asc;
```

Pregunta: en la sentencia anterior, qué debemos modificar para que el orden sea descendente y por nombre de compañía (no por tipo de avión).

Finalmente, si queremos complicarnos la vida y deseamos mostrar el código, el nombre del pasajero, el tipo de avión que utilizó y el asiento que ocupó *Harrison Ford*, podemos escribir la sentencia siguiente:

```
select Passenger.ID_psg, Passenger.name, Trip.plane, Pass_in_trip.place
from Trip join Pass_in_trip on Trip.trip_no = Pass_in_trip.trip_no
join Passenger on Passenger.ID_psg = Pass_in_trip.ID_psg
where Passenger.name = 'Harrison Ford';
```

En la sentencia anterior, nos piden mostrar información que se encuentran en tres tablas diferentes (*Trip*, *Passenger* y *Pass_in_trip*). Entonces, utilizamos dos cláusulas *JOIN* dentro de una misma cláusula *FROM*. Como podemos ver, cada sentencia *JOIN* necesita de la cláusula *ON* para que pueda funcionar correctamente.

Ejercicios

1. Mostrar el nombre de los pasajeros viajan a Paris o a Londres.
2. Mostrar el nombre de los pasajeros que prefieren sentarse al lado de la ventana (sitios "a" y "d").
3. Mostrar el lugar de destino de los pasajeros que tengan códigos con valores entre 10 y 20.
4. Mostrar todos los datos de los últimos 5 vuelos vendidas por nuestra empresa.
5. Mostrar los modelos de aviones únicos que realizaron un vuelo con nuestra empresa "Viajes Luca"
6. Mostrar el nombre, las ciudades de salida y destino de los pasajeros cuyo nombre empiecen con la letra "S".
7. Mostrar el código, el nombre del pasajero, el tipo de avión que utilizó en su viaje, las ciudades de salida y destino de los pasajero cuyo nombre empieza con la letra "S".

–
Hugo Alatrista-Salas
UP