SQL

Thursday, 19 March 2020 8:37 AM

- Structured Query Language
  - ↳ Database Query Language
    - Es de alto nivel
    - No procidimental
      - ↳ No tiene ciclos y condiciones*

ISO
ANCII } Standard de SQL

- Extensiones del lenguajes
  - PL-SQL ← Oracle
  - SQL-PL ← IBM

7.3. Categorías del SQL

El lenguaje SQL está dividido en diversas categorías clasificadas con base a su uso:

DDL (Data Definition Language)
Lenguaje de definición de datos. Es el lenguaje encargado de la creación, modificación y eliminación de la estructura de los objetos de la base de datos (tablas, índices, vistas, etc).

DML (Data Manipulation Language)
Lenguaje de manipulación de datos. Es el lenguaje que permite realizar las tareas de consulta, modificación y eliminación de los datos almacenados en una base de datos.

DCL (Data Control Language)
Lenguaje de control de datos. Es el lenguaje encargado de configurar y establecer el control de acceso a la base de datos. Incluye instrucciones para definir accesos y privilegios a los distintos objetos de la base de datos.

DQL (Data Query Language)
Lenguaje de consulta de datos. Algunos autores clasifican a la instrucción SELECT como el único elemento de una cuarta categoría del lenguaje SQL Data Query Language (DQL).

Transaction Control
Control de transacciones. Es el lenguaje empleado para crear, y administrar transacciones aplicadas a un conjunto de sentencias DML principalmente.

La siguiente tabla muestra las principales clausulas SQL divididas por categoría:

Lenguaje	Clausulas básicas

DDL (Data Definition Language)	`create`
	`alter`
	`drop`
	`rename`
	`truncate`
	`comment`

DML (Data Manipulation Language)	`insert`
	`update`
	`delete`
	`merge`

DCL (Data Control Language)	`grant` `revoke`
DQL (Data Query Language)	`select`

Transaction Control	`commit`
	`rollback`
	`savepoint`

- Si accedemos desde el usuario en sistema operativo no pide contraseña
- 3 Formas de acceder
  - Diccionario del DB
  - Archivo con contraseñas de usuarios
  - Sistema Operativo

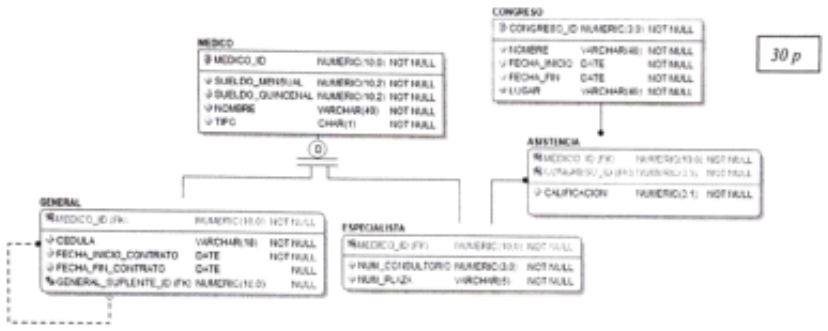
- USERS es el table space por defecto para guardar usuarios
- Los privilegios y permisos se le dan a los usuarios creados.

→ grant ≙ Dar permisos

REPASO

Tuesday, 5 May 2020 2:47 AM

1. *Considere el siguiente modelo relacional que guarda los datos de los médicos generales y especialistas de un hospital.* 45



30 p

a. *Generar el código SQL para crear las tablas* `medico`, `general` y `asistencia`. *Asumir que las tablas restantes ya fueron creadas. Considerar los siguientes requerimientos: Evitar cualquier posibilidad de inconsistencia en el campo sueldo quincenal (columna virtual). El sueldo mensual debe estar en el rango [8000, 88,999.99]. La cédula de los médicos no debe duplicarse, debe iniciar con el prefijo 'SEP'. La fecha inicio del contrato se le asigna la fecha del sistema en caso de no especificarse. La fecha fin del contrato debe ser por lo menos 3 meses después con respecto a la fecha inicio.*

b. *Generalmente se realizan búsquedas de los médicos generales empleando los últimos 5 caracteres de su cédula. Se requiere que las consultas sean lo más eficiente posible y evitar cargar la tabla completa y ejecutar repetidamente las funciones empleadas para extraer estos caracteres. Crear el(los) objeto(s) necesario(s) para realizar búsquedas eficientes.*

c. *Considerando el modelo completo, genere una lista de secuencias necesarias para implementar este modelo. Proponer un nombre para la secuencia y el nombre de la tabla donde se emplearía (no se requiere generar el código DDL).*

d. *Considerando el modelo completo, genere una lista de los índices que serán creados. Indicar nombre del índice, su tipo, tabla y campo(s) donde serán aplicados (no se requiere generar el código DDL).*

```
create table medico(
medico_id   number(10,0),
sueldo_mensual  number(10,2) not null,
sueldo_quincenal generated always as (sueldo_mensual/2)
 virtual,
nombre   varchar2(40) not null,
tipo     char(1) not null,
constraint  medico_pk primary key(medico_id),
constraint  sueldo_mensual_chk check
 check(sueldo_mensual between 8000 and 88999.99)
 );

create table general(
medico_id,
 cedula varchar2(18) not null,
 fecha_inicio_contrato date default sysdate not null,
```

```
fecha_inicio_contrato date default sysdate not null,
fecha_fin_contrato date,
general_suplente_id,
constraint general_medico_id_fk foreign key(medico_id)
 references medico(medico_id), // FK del padre
constraint general_pk primary key(medico_id), //Subtipos
constraint general_cedula_uk unique(cedula),
constraint general_cedula_chk check (cedula like 'SEP%')
constraint general_fecha_fin_chk
 check( MONTHS_BETWEEN(fecha_fin, fecha_inicio)>=3),
 constrant general_general_id_fk foreign key (general_suplente_id)
  references general(medico_id)
);

create table asistencia(
medico_id,
congreso-id,
calificacion number(3,1) not null ,
 constraint asistencia_medico_id foreign key(medico_id)
 references especialista(medico_id),
constraint asistencia_congreso_id foreign key (congreso_id),
 references congreso(congreso_id),
constraint asistencia_pk primary key (medico_id, congreso_id)
);
```

← No olvidar poner ambas

b
- Index → Índices ≙ Para búsquedas
- Secuencias → Para los id's

```
create index general_cedula_ix on
 general (SUBSTR(cedula,-1,5));
```

← columna

create index general_cedula_ix on
general (SUBSTR(cedula, -1, 5));

Tabla — columna — condición

create index emp_empleados_ix on
empleado_indexado (nombre, ...,);

columna(s)

c) Sintaxis Secuencias

```
create sequence seq_medico
start with 500
increment by 10
max value 600
min value 0
nocycle;
```

.currval
.nextval

seq_medico en medico

seq_congreso en congreso

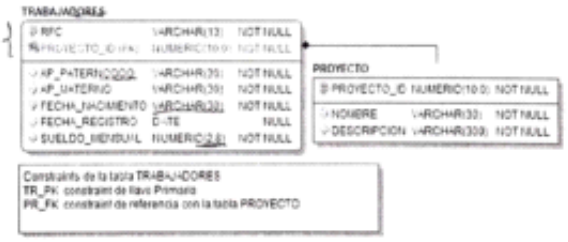
d) ▶ Cada PK tiene un índice implícito
▶ Se crear explícitamente índices

- Unique → Sólo una columna
- Compuestos → Varias columnas
- Nonunique → ¿Qué usan funciones?

general_cedula_ix	explicito, nonunique
medico_pk	implicito, unique
congreso_pk	...
general_pk	...
asistencia_pk	implicito, compuesto

2. *El siguiente modelo relacional muestra las tablas de una base de datos ya creada donde se almacena la información de los trabajadores y su proyecto asignado.*

El DBA ha detectado varios problemas de diseño. Genere las sentencias SQL necesarias en el orden correcto para corregir los errores de tal forma que los siguientes datos considerados como correctos puedan insertarse sin problemas.




```
insert into trabajador(
   trabajador_id, rfc,
   nombre, ap_paterno,ap_materno,
   fecha_nacimiento,fecha_registro,sueldo_mensual,proyecto_id)
values(1,'ROMND34209LMZ', 'Juan', 'Lara',null,
   to_date('02/02/1979','dd/mm/yyyy'),default,35420.55,1);
```

to_date('-,-,-','dd/mm/yyyy')

// Renombrar tabla

```
alter table trabajadores rename to trabajador;
```

// Renombrar columna

```
alter table trabajador rename column
ap_paternooo to ap_paterno;
```

// Cambiar tipo de dato

```
alter table trabajador modify fecha_nacimiento date;
```

// Como hay un default establecerlo

```
alter table trabajador modify fecha_registro default sysdate;
```

// Corregir number

```
alter table trabajador modify sueldo_mensual number(8,2);
```

// Agregar columna

```
alter table trabajador add nombre varchar2(50)
not null;
```

```
alter table trabajador modify ap_materno null;
```

// Cambiar PK

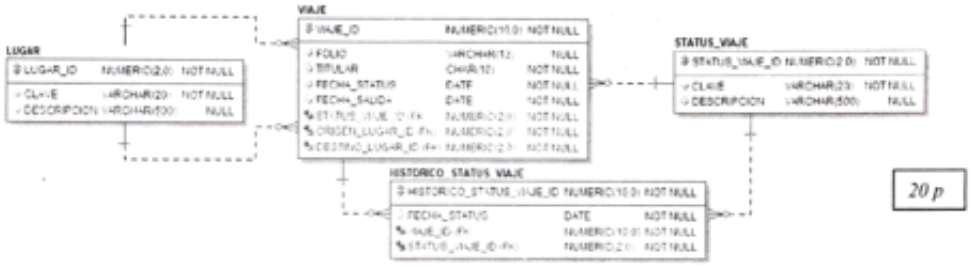
```
alter table trabajador drop constraint TR_PK;
alter table trabajador add trabajador_id
number(10,0) constraint tr_pk primary key;
```

---

3. *Considerar el siguiente modelo relacional para contestar cada una de las siguientes preguntas.*

a. *Se ha detectado que los viajes con fecha de salida en marzo del 2010 fueron capturados de forma incorrecta por lo que se pide ejecutar una sola sentencia SQL que permita corregir los siguientes errores. El folio del viaje no fue registrado. Su valor debe estar formado por los 3 primeros caracteres del titular en mayúsculas seguido del identificador del viaje. El status del viaje debe ser CONCLUIDO. la fecha del status se recorrió un día hacia adelante lo cual es incorrecto. los valores de origen y destino se capturaron en orden inverso.*

b. *Suponer la existencia de una tabla llamada* `h_respaldo`. *Dicha tabla debe ser actualizada periódicamente con los valores del histórico de status. Genere una sentencia SQL que permita realizar dicha sincronización.*



20 p

a)

```
update viaje
    set folio = (UPPER(SUBSTR(titular,1,3))
                 || to_char(viaje_id)),

    status_viaje_id = ( select status_viaje_id
    from status_viaje
    where clave = 'CONCLUIDO'),
    fecha_status = fecha_status - 1,
    origen_lugar_id = destino_lugar_id,
    destino_lugar_id = origen_lugar_id

where to_char(fecha_salida,'MM,YYYY') = '03/2010';
```

b)

Tabla respaldo

Tabla origen datos

```
merge into h_respaldo hr
using historico_status_viaje h
on (hr.historico_status_viaje_id =
    h.historico_status_viaje_id)
```

```
when matched then
    update set hr.fecha_status = h.fecha_status,
               hr.viaje_id = h.viaje_id,
               hr.status_viaje_id = h.status_viaje_id

when not matched then
    insert(hr.historico_status_id, hr.fecha_status,
           hr.viaje_id, hr.status_viaje_id)
    values(h.historico_status_id, h.fecha_status,
           h.viaje_id, h.status_viaje_id);
```

---

4. *Considerar que la base de datos ha sido configurada con el nivel de aislamiento* read commited, *así como los datos de la tabla* prod *la cual contiene los identificadores de los productos y su precio. Revisar la siguiente secuencia de eventos provocados por 2 transacciones. Responder las preguntas en cada caso. Justificar respuesta.*

id	Precio
1	100
2	150
3	200

T	Sesión 1	Sesión 2
1	`select precio from prod where id = 3`	
2		`select precio from prod where id =3` P1: ¿Qué se obtendrá?, Justificar respuesta
3	`update prod set precio = 1000 where id =2`	
4		`update prod set precio = 2000 where id = 1` P2: ¿Qué se obtendrá?, Justificar
5	`update prod set precio = 5000 where id = 1` P3: ¿Qué se obtendrá? Justificar.	
6		`select precio from prod where id =1` P4: ¿Qué se obtendrá?, Justificar
7		`rollback;`
8		`Select precio from prod where id =1` P5: ¿Qué se obtendrá? Justificar

P1: 200, no hay ningún cambio en ninguna sesión, es sólo una consulta

P2: El id=1 se actualizará a 2000 en la sesión 2

P3: Se bloquea la sesión 1 en T3 debido a que T2 está usando ese registro (No ha finalizado)

P4: Muestra el valor de 2000 debido a que es el actual en la transacción

P4: Muestra el valor de 2000 debido a que es el actual en la transacción

P5: Muestra 100, debido a que el rollback se deshacen todos los cambios en la sesión 2 y como la sesión 1 no ha hecho commit (No ha finalizado) se queda el id=1 con el valor inicial.

* Alter, commit, exit → Termina transacción
* rollback → Te regresas