Evaluaciones de SQL 2022

Parcial

1- inner join

Enunciado

Listado de tour contratados. Listar para cada tour contratado entre enero y julio de 2022 (inclusive) los clientes que lo contrataron. Indicar número y temática del tour, fecha y hora de contratación y cuil, tipo y denominación del cliente. No se deberán mostrar registros repetidos y deberá ordenarse alfabéticamente por denominación del cliente.

Resolución

```
select t.nro, t.tematica, co.fecha_hora
      , cli.cuil, cli.tipo, cli.denom
from contrata co
inner join tour t
      on co.nro_tour = t.nro
inner join cliente  cli
      on co.cuil_cliente=cli.cuil
where co.fecha_hora between '20220101' and '20220731'
order by cli.denom;
```

#Nota: opcionalmente pueden usar un distinct pero validar que no hagan mal el join

2- left/right join

Enunciado

Idiomas y solicitudes de julio de 2022. Listar los idiomas y, para aquellos que hayan sido solicitados en contrataciones de julio de 2022, mostrar en qué tours fueron solicitados. Indicar código y nombre del idioma, número de tour y temática del tour. Ordenar por nombre del idioma alfabéticamente

Resolución

```
select i.codigo, i.nombre
      , t.nro, t.tematica
from idioma i
left join contrata con
      on i.codigo = con.codigo_idioma
    and con.fecha_hora between '20220701' and '20220731'
left join tour t
      on con.nro_tour=t.nro
order by i.nombre
```

3 - group by/having

Enunciado

Actividades poco frecuentes. Listar las actividades que se hayan realizado menos de 3 veces en 2022. Indicar código de la locación, número y descripción de la actividad, cuántas veces se realizó en 2022, última vez que se realizó y la cantidad de días que pasaron desde que se realizó por última vez hasta hoy. Ordenar por cantidad de veces descendente y cantidad de días desde que se realizó por última vez ascendente.

Resolución

```
select a.codigo_locacion, a.nro, a.descripcion
    , count(e.codigo_locacion) cant
    , max(e.fecha_hora_ini) ultima_vez
    , datediff(now(),(max(e.fecha_hora_ini))) dias_desde_ultima_vez
from actividad a
left join escala e
    on a.codigo_locacion=e.codigo_locacion
    and a.nro=e.nro_actividad
    and fecha_hora_ini between '20220101' and '20221231'
group by a.codigo_locacion, a.nro, a decripcion
having cant < 3
order by cant desc, dias_desde_ultima_vez asc;
```

4 - diferencia

Enunciado

Tours particulares del 2022. Listar todos los tours realizados en 2022 que no hayan sido contratados por algún cliente revendedor o empresa de turismo. Indicar nro de tour, fechas de salida y regreso y precio unitario sugerido.

Resolución

```
select t.nro, t.fecha_hora_salida, t.fecha_hora_regreso,
t.precio_unitario_sugerido
from tour t
where t.fecha_hora_salida between '20220101' and '20221231'
and t.nro not in (
    select con.nro_tour
    from contrata con
    inner join cliente cli
        on cli.cuil=con.cuil_cliente
    where cli.tipo in ('revendedor', 'empresa turismo')
```

```
);
```

5 - subcons/tt/cte/var

Enunciado

Guías en declive. Listar guías que hayan guiado menos tours en 2022 que en 2021. Se deberán mostrar incluso aquellos guías que no hayan guiado tours en 2021 o 2022. Indicar cuil, nombre, apellido y categoría del guía, cantidad de tours guiados en 2022, cantidad de tours guiados en 2021 y declive en la cantidad.

Resolución

```
with cant2022 as (
    select guia.cuil, guia.nombre, guia.apellido, guia.categoria
        , count(tour.cuil_guia) cant
    from empleado guia
    left join tour
        on guia.cuil=tour.cuil_guia
        and tour.fecha_hora_salida between '20220101' and '20221231'
    where guia.tipo='guia'
    group by guia.cuil, guia.nombre, guia.apellido, guia.categoria
)
select guia.cuil, guia.nombre, guia.apellido, guia.categoria
    , count(tour.cuil_guia) cant2021
    , coalesce(cant2022.cant,0) cant2022
    , count(tour.cuil_guia) - coalesce(cant2022.cant,0) declive
from empleado guia
left join tour
    on guia.cuil=tour.cuil_guia
    and tour.fecha_hora_salida between '20210101' and '20211231'
left join cant2022
    on guia.cuil=cant2022.cuil
where guia.tipo='guia'
group by guia.cuil, guia.nombre, guia.apellido, guia.categoria
having cant2022 < cant2021;
```

6 - transacción

Enunciado

Transferir actividades a nueva locación. La empresa ha adquirido una nueva locación para tours para utilizar cierto equipamiento. Se deberá dar de alta la nueva locación con los datos que figuran a continuación y transferir todas las actividades de locaciones con

ambientación “SNK” y que utilicen el equipamiento “Equipo de maniobras tridimensionales” a esta nueva locación.

Nueva locación:
Nombre: Shiganshina District
Ambientación: SNK
Ubicación GPS: POINT(25.083333, -77.333333)
Dirección: Paradis 2013

Resolución

```
begin;

insert into locacion (nombre, ambientacion, ubicacion_gps,
direccion)
values('Shiganshina District', 'SNK', POINT(25.083333, -77.333333),
'Paradis 2013');

select last_insert_id() into @sd;

update actividad
set codigo_locacion = @sd
  , nro=@act:=@act+1 #totalmente opcional
where equipamiento='Equipo de maniobras tridimensionales'
    and codigo_locacion in (
        select codigo
        from locacion
        where ambientacion='SNK'
    );

commit;
```

7 - sp/func/trigger

Enunciado

Crear una función prop_tematica que reciba una temática y un rango de fecha y hora (desde y hasta) y calcule la proporción de cantidad de entradas contratadas por tours con dicha temática en el rango propuesto sobre el total de entradas contratadas en el mismo rango.
Listar todas las temáticas y para cada una invocar a la función para calcular la proporción entre agosto y diciembre de 2022. Indicar tematica y proporción ordenar por proporción descendente. No se deben repetir temáticas.

Resolución

```
USE `role_play_events`;
DROP function IF EXISTS `prop_tematica`;

DELIMITER $$
USE `role_play_events`$$
CREATE DEFINER=`gestiondatos`@`%` FUNCTION `prop_tematica`(tema
varchar(255), desde datetime, hasta datetime) RETURNS decimal(10,3)
    READS SQL DATA
BEGIN
declare prop decimal(10,3);
declare tot decimal(10,3); #puede ser int, uso dec para asegurar el
tipo del resultado final

select sum(con.cant_entradas) into tot
from contrata con
where con.fecha_hora between desde and hasta;

select sum(con.cant_entradas)/tot into prop
from contrata con
inner join tour t
on con.nro_tour=t.nro
where con.fecha_hora between desde and hasta
and t.tematica=tema;


RETURN prop;
END$$

DELIMITER ;

# invocación
select distinct t.tematica,
prop_tematica(t.tematica,'20220801','20221231') prop
from tour t
order by prop desc
```

Recuperatorio

1 - inner join

Enunciado

Actividades de una temática. Listar las temáticas de los tours y todas las actividades que se han realizado en las escalas de los tours de dicha temática que comenzaron durante el 2021. Indicar, temática, código y nombre de la locación, número, descripción y equipamiento de la actividad. Ordenar por temática alfabéticamente. No se deben repetir registros.

Resolución

```
select distinct t.tematica
    , l.codigo, l.nombre
    , a.nro, a.descripcion, a.equipamiento
from tour t
inner join escala e
    on t.nro=e.nro_tour
inner join actividad a
    on e.codigo_locacion=a.codigo_locacion
    and e.nro_actividad=a.nro
inner join locacion l
    on a.codigo_locacion=l.codigo
where t.fecha_hora_salida between '20210101' and '20211231T235959'
order by t.tematica;
```

2 - left/right join

Enunciado

Locaciones visitadas por tours. Indicar todas las locaciones con ambientación SNK y si fueron visitadas durante un tour en agosto de 2022 indicar los datos del mismo, la escala y actividad por la cual la visitaron. Indicar código y nombre de la locación, número, fecha de salida y temática del tour, número de la actividad y fecha y horas de la escala.

Resolución

```
select l.codigo, l.nombre
    , t.nro, t.fecha_hora_salida, t.tematica
    , e.nro_actividad, e.fecha_hora_ini, e.fecha_hora_fin
from locacion l
left join escala e
```

```
    on l.codigo=e.codigo_locacion
    and e.fecha_hora_ini between '20220801' and '20220831T235959'
left join tour t
    on e.nro_tour=t.nro
where l.ambientacion='SNK';
```

3 - group by/having

Enunciado

Empresas de turismo con pocos contratos en 2022. Listar las empresas de turismo que hayan contratado menos de 3 tours distintos en 2022. Indicar cuil y denominación del cliente, cantidad de tours distintos contratados, cantidad total de entradas y fecha del último contrato. Ordenar por cantidad total de entradas descendente y fecha del último contrato ascendente.

Resolución

```
select cli.cuil, cli.denom
    , count(distinct con.nro_tour) cant_tour_dist
    , sum(con.cant_entradas) total_entradas
    , max(fecha_hora) ult_fecha_cont
from cliente cli
left join contrata con
    on cli.cuil=con.cuil_cliente
    and con.fecha_hora between '20220101' and '20221231T235959'
where cli.tipo='empresa de turismo'
group by cli.cuil
having  cant_tour_dist < 3
order by total_entradas desc, ult_fecha_cont;
```

4 - diferencia

Enunciado

Locaciones “.hack//SIGN” poco utilizadas. Listar las locaciones de ambientación “.hack//SIGN” que no hayan sido utilizadas en alguna escala que haya comenzado desde junio hasta la fecha de hoy. Indicar código y nombre de la locación.

Resolución

```
select l.codigo, l.nombre
from locacion l
where l.ambientacion='.hack//SIGN'
     and l.codigo not in (
```

```
            select e.codigo_locacion
        from escala e
        where e.fecha_hora_ini between '20220601' and now()
);
```

5 - subcons/TT/CTE/var

Enunciado

Actividades con reducción de costos. Listar todas las actividades cuyo valor actual sea menor al anterior. En caso de no tener un valor anterior no deben listarse. Indicar número, descripción y equipamiento de la actividad, costo actual, costo anterior y diferencia. Ordenar por valor actual ascendente y diferencia descendente.

Resolución

```
with val_act as (
    select c.codigo_locacion, c.nro_actividad, max(c.fecha_desde)
ult_fec
    from costo c
    where c.fecha_desde < now()
    group by c.codigo_locacion, c.nro_actividad
), val_ant as (
    select c.codigo_locacion, c.nro_actividad, max(c.fecha_desde)
ult_fec
    from costo c
    inner join val_act va
           on c.codigo_locacion=va.codigo_locacion
        and c.nro_actividad=va.nro_actividad
    where c.fecha_desde < va.ult_fec
    group by c.codigo_locacion, c.nro_actividad
)
select a.nro, a.descripcion, a.equipamiento
    , c_act.valor costo_actual, c_ant.valor costo_anterior,
c_ant.valor - c_act.valor diff
from actividad a
inner join val_act
    on a.codigo_locacion=val_act.codigo_locacion
    and a.nro=val_act.nro_actividad
inner join costo c_act
     on c_act.codigo_locacion=val_act.codigo_locacion
    and c_act.nro_actividad=val_act.nro_actividad
inner join val_ant
```

```
           on a.codigo_locacion=val_ant.codigo_locacion
        and a.nro=val_ant.nro_actividad
inner join costo c_ant
      on c_ant.codigo_locacion=val_ant.codigo_locacion
    and c_ant.nro_actividad=val_ant.nro_actividad
where c_act.valor < c_ant.valor
order by costo_actual, diff desc
```

6 - transacción

Enunciado

Promoción y reemplazo. El encargado “Bertolt Hoover” pasó a ser guía y se contrató un nuevo empleado para reemplazarlo en las escalas. Se debe dar de alta el nuevo empleado con los datos que figuran a continuación, reasignar las escalas posteriores al 30 de noviembre asignadas a “Bertolt Hoover” al nuevo encargado. Cambiar el tipo de “Bertolt Hoover” de encargado a guía y asígnarle los tours con temática “SNK” que comiencen posteriores al 30 de noviembre.

Nuevo empleado:
CUIL: 85-85858585-8
Nombre: Armin
Apellido: Arlert
Teléfono: +858-585858585
Categoría: comander
Tipo: encargado

Resolución

```
begin;

select cuil into @bh
from empleado
where nombre='Bertolt'
    and apellido='Hoover';

insert into empleado
values(85858585858, 'Armin', 'Arlert',
'+858-585858585','comander','encargado');

update escala
set cuil_encargado=85858585858
where cuil_encargado=@bh
and fecha_hora_ini > '20221130';
```

```
update empleado
set tipo='Guia'
where cuil=@bh;

update tour
set cuil_guia=@bh
where tematica='SNK'
    and fecha_salida > '20221130';

commit;
```

7 - sp/func/trigger

Enunciado

Crear una función llamada idioma_principal que reciba como parámetro una temática e informe el código idioma más solicitado por cantidad de contratos para dicha temática. Listar para las temáticas de los tours iniciados en 2022, el idioma principal de cada una invocando a la función. Indicar, temática, código mediante la función y nombre del idioma principal, ordenar por temática ascendente. No se deben repetir temáticas.

Resolución

```
DELIMITER $$
CREATE DEFINER=`gestiondatos`@`%` FUNCTION `idioma_principal`(tema
varchar(255)) RETURNS varchar(255) CHARSET utf8mb4
    READS SQL DATA
BEGIN
declare cip varchar(255);

select t.nro into cip
from tour t
inner join contrata c
     on t.nro=c.nro_tour
where t.tematica=tema
group by t.nro
order by count(*) desc
limit 1;

RETURN cip;
END$$
DELIMITER ;
```

```
select distinct t.tematica, i.codigo, idioma_principal(t.tematica)
from tour t
inner join idioma i
      on i.codigo= idioma_principal(t.tematica)
where t.fecha_hora_salida between '20220101' and '20221231T235959'
order by t.tematica
```

Globalizador

1 - inner join

Enunciado

Empleados que trabajaron en tours de SNK. Listar todos los empleados que trabajaron en algún tour con temática SNK ya sea como guías o como encargados de alguna de sus escalas. Indicar cuil, nombre y apellido del empleado, número, fecha de salida del tour. Ordenar por apellido y nombre alfabéticamente, fecha y hora de salida ascendente y no repetir registros

Resolución

```
select g.cuil, g.nombre, g.apellido,t.nro, t.fecha_hora_salida
from tour t
inner join empleado g
on t.cuil_guia=g.cuil
where t.tematica='SNK'
union
select enc.cuil, enc.nombre, enc.apellido,t.nro, t.fecha_hora_salida
from tour t
inner join escala e
on t.nro=e.nro_tour
inner join empleado enc
on e.cuil_encargado=enc.cuil
where t.tematica='SNK'
order by apellido, nombre, fecha_hora_salida desc;
```

2 - left join

Enunciado

Guías y contratos de Lord of the Rings. Listar todos los guías de la empresa y si han guiado algún tour con temática “Hunter X” la información de contratos. Indicar cuil, nombre y apellido del guía y si hay número de tour, año y mes de salida, importe y cantidad de entradas del contrato, cuil y denominación del cliente y si se solicitó algún idioma código y nombre del mismo.

Resolución

```
select g.cuil cuil_guia, g.nombre, g.apellido
     , t.nro, year(t.fecha_hora_salida) anio,
month(t.fecha_hora_salida) mes
```

```
    , c.importe, c.cant_entradas
    , cli.cuil cuil_cliente, cli.denom
    , i.codigo codigo_idioma, i.nombre idioma
from empleado g
left join tour t
     on g.cuil=t.cuil_guia
     and t.tematica='hunter x'
left join contrata c
     on t.nro=c.nro_tour
left join idioma i
     on c.codigo_idioma=i.codigo
left join cliente cli
     on c.cuil_cliente=cli.cuil
where g.tipo='guia';
```

3 - group by/having

Enunciado

Clientes particulares que disfrutaron alguna temática en 2021. Listar los clientes particulares que en 2021 hayan contratado más de 5 tours de una misma temática. Indicar cuil, denominación del cliente, temática, cantidad de tours distintos contratados, cantidad total de entradas y fecha del último contrato de dicha temática. Ordenar alfabéticamente por denominación del cliente y fecha de último contrato descendente.

Resolución

```
select cli.cuil, cli.denom , t.tematica
     , count(distinct t.nro) cant_tours, sum(con.cant_entradas)
tot_entradas, max(con.fecha_hora) ult_cont
from contrata con
inner join cliente cli
     on con.cuil_cliente=cli.cuil
inner join tour t
     on con.nro_tour=t.nro
where con.fecha_hora between '20220101' and '20211201'
group by cli.cuil, t.tematica
having cant_tours >5
order by cli.denom, ult_cont desc;
```

4 - diferencia

Enunciado

Guías que hablen idiomas no solicitados en 2022. Listar los guías que hablen idiomas que no hayan sido solicitados para ningún contrato de 2022. Indicar cuil, nombre y apellido del empleado, código y nombre del idioma. Ordenar por nombre del idioma alfabético y cuil ascendente.

Resolución

```
select g.cuil, g.nombre, g.apellido
      , idi_no_sol.codigo, idi_no_sol. nombre
from idioma idi_no_sol
inner join idioma_guia ig
      on idi_no_sol.codigo=ig.codigo_idioma
inner join empleado g
      on g.cuil=ig.cuil_guia
where idi_no_sol.codigo not in
           (
                select idi_solic.codigo_idioma
            from contrata idi_solic
            where idi_solic.fecha_hora between '20220101' and
'20221201'
           )
order by idi_no_sol.nombre, g.cuil;
```

5 - subcons/TT/CTE/var

Enunciado

Empleados mejor pagados. Listar aquellos empleados cuyo salario por hora actual sea superior al promedio de los salarios actuales de todos los empleados. Indicar cuil, nombre, apellido y categoría del empleado, su salario actual y el promedio de los salarios actuales. Ordenar por salario actual descendente y por apellido y nombre alfabéticamente.

Resolución

```
with val_fec as (
      select sh.cuil_empleado, max(sh.fecha_desde) ult_fec
    from salario_hora sh
    where sh.fecha_desde ⩽ now()
    group by sh.cuil_empleado
) , sal_act as (
select sal.cuil_empleado, sal.fecha_desde, sal.valor
```

```
from val_fec
inner join salario_hora sal
      on sal.cuil_empleado=val_fec.cuil_empleado
     and sal.fecha_desde=val_fec.ult_fec
)
select e.cuil, e.nombre, e.apellido, e.categoria
      , sal_act.valor salario_actual, (select avg(valor) from
sal_act) promedio
from sal_act
inner join empleado e
on sal_act.cuil_empleado=e.cuil
where sal_act.valor > (select avg(valor) from sal_act)
order by salario_actual desc, apellido, nombre;
```

6 - transacción

Enunciado

Duplicar tour. El tour 5757 tuvo mucho éxito. Se debe dar de alta un nuevo tour 6060 con los mismos datos y las mismas escalas a realizarse 4 meses más tarde tanto para el tour como sus escalas y en lugar del guía de 5757, asignarle el guía que menos tours haya guiado hasta ahora.

Resolución

```
begin;

with guias_tour as (
    select cuil_guia, count(*) cant
    from tour
    group by cuil_guia
)
select cuil_guia into @guiapoco
from guias_tour
where cant = (select min(cant) from_guias_tour)
limit 1;

insert into tour
select 6060, ADDDATE(fecha_hora_salida, INTERVAL 4 MONTH),
ADDATE(fecha_hora_regreso, INTERVAL 4 MONTH), lugar_salida
  , precio_unitario_sugerido, vehiculo, tematica, @guiapoco
from tour
where nro=5757;
```

```
insert into escala
select 6060, ADDDATE(fecha_hora_ini, INTERVAL 4 MONTH)
    , ADDDATE(fecha_hora_fin, INTERVAL 4 MONTH), codigo_locacion
    , nro_actividad, cuil_encargado
from escala
where nro_tour=5757;

commit;
```

7 - sp/func/trigger

Enunciado

Crear un store procedure baja_asistencia que reciba un rango de fechas (desde y hasta) y liste las temáticas de los tours iniciados entre dicho rango para las que la cantidad total de asistentes fue menor al 60% de la cantidad total de entradas contratadas. Indicar temática, fecha de salida del último tour realizado en dicha temática entre el rango de fechas, cantidad total de entradas contratadas y cantidad de asistentes y porcentaje de asistencia. Ordenar por porcentaje de asistentes ascendente y temática del tour alfabético.

Invocar el procedimiento para los tours iniciados entre 120 y 30 días atrás.

Resolución

```
DROP procedure IF EXISTS `baja_asistencia`;
DELIMITER $$

CREATE DEFINER=`gestiondatos`@`%` PROCEDURE `baja_asistencia`(in
since_date date, in until_date date)
BEGIN

select t.tematica, max(t.fecha_hora_salida) ultima_salida
     , sum(con.cant_entradas) total_contratado,
sum(asistencia.cant_asist) total_asist
    , 100*sum(asistencia.cant_asist)/sum(con.cant_entradas)
porcen_asist
from tour t
inner join contrata con
     on t.nro=con.nro_tour
inner join
     (
          select ac.nro_tour, count(*) cant_asist
          from asistente_contrato ac
```

```
            group by ac.nro_tour
      ) asistencia
    on t.nro=asistencia.nro_tour
where t.fecha_hora_salida between since_date and until_date
group by t.tematica
having porcen_asist < 60
order by porcen_asist, t.tematica;

END$$

DELIMITER ;

call
baja_asistencia(adddate(current_date,-120),adddate(current_date,-30)
)
```

Parcial AD

Ej 1

Enunciado

****Sugerencias de tours.**** Para una promoción se desea sugerir a cada asistente otros tours que sean de su interés. Para cada asistente que haya realizado más de un tour en alguna temática, identificar la temática más frecuente elegida por el asistente y sugerirle todos los tours futuros con la misma temática. Indicar dni, nombre, apellido y teléfono del asistente y de los tours, número, temática, fecha y hora de salida y precio unitario. Ordenar por apellido y nombre alfabético y fecha y hora de salida descendente.

Resolución

```mysql
#### Opción A)

with asist_tema  as (
     select dni_asistente, t.tematica, count(*) cant
     from asistente_contrato ac
     inner join tour t
          on ac.nro_tour=t.nro
     group by dni_asistente, t.tematica
    having cant > 1
    order by count(*) desc
),
max_asist as (
     select dni_asistente, max(cant) max_cant
    from asist_tema
    group by dni_asistente
)
select a.dni, a.apellido, a.nombre, a.telefono
          , t.nro, t.tematica, t.fecha_hora_salida,
t.precio_unitario_sugerido
from asist_tema ate
inner join max_asist ma
     on ate.dni_asistente=ma.dni_asistente
    and ate.cant=ma.max_cant
left join tour t #se perdona un inner
     on t.tematica=ate.tematica
    and t.fecha_hora_salida>now()
inner join asistente a
```

```
     on ate.dni_asistente=a.dni
order by a.apellido, a.nombre, t.fecha_hora_salida desc;
```

```mysql
#### Opción B)

select a.dni, a.apellido, a.nombre, a.telefono
          , t.nro, t.tematica, t.fecha_hora_salida,
t.precio_unitario_sugerido
from asistente a
left join tour t #se perdona un inner
on t.tematica=(
    select tour.tematica
    from asistente_contrato ac
    inner join tour
           on ac.nro_tour=tour.nro
    where ac.dni_asistente=a.dni
    group by tour.tematica
    having count(*) > 1
    order by count(*) desc
    limit 1
)
and  t.fecha_hora_salida > now()
order by a.apellido, a.nombre, t.fecha_hora_salida desc;
```

Ej 2

Enunciado

****Determinar el margen de ganancia por temática**.** Crear una función que dado un tour devuelva ~~la~~ el costo total de todas sus actividades a la fecha de salida y otra función que dado un tour devuelva el ingreso utilizando los contratos. Finalmente utilizando dichas funciones calcular el margen de ganancia total de cada temática para los cursos ya iniciados. Indicar, temática, cantidad de tours y margen de ganancia. Ordenar por margen de ganancia descendente.

Margen de ganancia: (ingresos - costos)/costos

Resolución

```mysql
USE `role_play_events`;
DROP function IF EXISTS `costo_actividades_tour`;

USE `role_play_events`;
DROP function IF EXISTS `role_play_events`.`costo_actividades_tour`;
;

DELIMITER $$
USE `role_play_events`$$
CREATE DEFINER=`gestiondatos`@`%` FUNCTION
`costo_actividades_tour`(nro_t int unsigned) RETURNS decimal(10,3)
    READS SQL DATA
BEGIN
declare costo_total decimal(10,3);
with val_fec as (
     select c.codigo_locacion, c.nro_actividad, max(c.fecha_desde)
fec_val
    from costo c
    inner join escala e
           on e.codigo_locacion=c.codigo_locacion
        and e.nro_actividad=c.nro_actividad
    where c.fecha_desde ⩽(select t.fecha_hora_salida from tour t
where t.nro=nro_t)
    and e.nro_tour=nro_t
    group by c.codigo_locacion, c.nro_actividad
)
select sum(c.valor) into costo_total
from val_fec
inner join costo c
  on c.codigo_locacion=val_fec.codigo_locacion
  and c.nro_actividad=val_fec.nro_actividad;

RETURN costo_total;
END$$


CREATE FUNCTION `ingreso_contrato_tour` (nro_t int unsigned)
RETURNS decimal(10,3)
```

```
reads sql data
BEGIN
declare ingreso_tot decimal (10,3);

select sum(c.importe) into ingreso_tot
from contrata c
where c.nro_tour=nro_t;

RETURN ingreso_tot;
END$$

DELIMITER ;

select t.tematica,
sum((coalesce(ingreso_contrato_tour(t.nro),0)-coalesce(costo_activid
ades_tour(t.nro),0))/coalesce(costo_actividades_tour(t.nro),1))
ganancia
from tour t
where t.fecha_hora_salida<now()
group by t.tematica;
```

Ej 3

Enunciado

****Consistencia de datos.**** La empresa notó que tanto la temática del tour como la ambientación de locación son en realidad lo mismo. Se decidió crear la tabla tematica (identificada por un código autoincremental y con una descripción), cargar las descripciones con los datos de tour y ambientación y modificar las tablas tour y locación para migrar las viejas columnas por la correspondiente clave foránea a la tabla temática.

Resolución

```mysql
use role_play_events;
create table tematica (
  codigo int unsigned not null auto_increment,
  descripcion varchar(255) not null,
  primary key (codigo));
```

```
alter table tour
add column cod_tematica int unsigned null,
add constraint fk_tour_tematica foreign key(cod_tematica) references
tematica(codigo) on delete restrict on update cascade;

alter table locacion
add column cod_tematica int unsigned null,
add constraint fk_locacion_tematica foreign key(cod_tematica)
references tematica(codigo) on delete restrict on update cascade;

begin;

insert into tematica(descripcion)
select distinct tematica from tour
union
select distinct ambientacion from locacion;

update tour
inner join tematica
     on tour.tematica=tematica.descripcion
set tour.cod_tematica=tematica.codigo;

update locacion
inner join tematica
     on locacion.ambientacion=tematica.descripcion
set locacion.cod_tematica=tematica.codigo;

commit;

alter table tour
modify column cod_tematica int unsigned not null,
drop column tematica;


alter table locacion
modify column cod_tematica int unsigned not null,
drop column ambientacion;
```

Recuperatorio AD

Ej 1

Enunciado

****Sugerir encargados a promover para guía.**** Debido al incremento en la demanda se deben nombrar nuevos guías para los tours de entre los encargados. Mostrar para cada temática el encargado que más escalas ya finalizadas ha coordinado en tours de dicha temática. Solo se deberán sugerir encargados que no hayan sido asignados como guías a ningún tour hasta ahora. Indicar temática, cuil, nombre y apellido del empleado y cantidad de escalas coordinadas. Ordenar por cantidad de escalas coordinadas descendente y temática alfabéticamente.

Resolución

```mysql
with cant_enc as (
    select t.tematica, e.cuil_encargado, count(e.cuil_encargado)
cant
    from tour t
    inner join escala e
        on t.nro=e.nro_tour
    where e.fecha_hora_fin < now()
    and e.cuil_encargado not in (
        select tr.cuil_guia
        from tour tr
    )
    group by t.tematica, e.cuil_encargado
),
max_cant as (
    select ce.tematica, max(cant) max_cant
    from cant_enc ce
    group by ce.tematica
)
select ce.tematica, ce.cuil_encargado
    , e.nombre, e.apellido, ce.cant
from max_cant mc
inner join cant_enc ce
    on mc.tematica=ce.tematica
    and mc.max_cant=ce.cant
inner join empleado e
    on ce.cuil_encargado=e.cuil
```

```
order by ce.cant desc, ce.tematica;
```

Ej 2

Enunciado

****Cálculo de costo del personal por tour.**** Crear la función costo_hora_empleado que reciba el cuil y una fecha y devuelva el salario por hora a esa fecha.
Luego crear un SP costo_tours que reciba una fecha y hora y liste todos los tours que ya sucedieron (fecha y hora de regreso menor o igual dicha fecha y hora) indicando los costos del guía y la sumatoria de los costos de los encargados invocando a la función antes creada. Indicar número, temática, fecha y hora de salida y regreso del tour, costo del guía a la fecha y hora de salida, sumatoria del costo de los encargados a la fecha y hora de inicio de cada escala y la suma de ambos costos. Ordenar por costo total descendente.
Finalmente invocar el SP con la fecha y hora actual.

Costo guia: horas_trabajadas_tour * salario_fecha_salida_tour
horas_trabajadas_tour: diferencia en horas entre fecha_hora_salida y fecha_hora_llegada.

Costo encargado: horas_trabajadas_escala* salario_fecha_hora_ini_escala
horas_trabajadas_escala: diferencia en horas entre fecha_hora_ini y fecha_hora_fin.

****Nota:**** Si no conoce otra función o forma que le resulte familiar para los cálculos de tiempo, puede usar la función `**timestampdiff(time_unit, datetime_since, datetime_until)**` que permite calcular la diferencia de tiempo en una unidad por ejemplo:
select timestampdiff(hour,'2022-01-01',now());

Resolución

```mysql
DROP function IF EXISTS `costo_hora_empleado`;
DROP procedure IF EXISTS `costo_tours`;

DELIMITER $$

CREATE DEFINER=`gestiondatos`@`%` FUNCTION
`costo_hora_empleado`(cuil_guia bigint, fecha_valor datetime)
RETURNS int
    READS SQL DATA
BEGIN

declare valor_hora decimal(10,3);
```

```
select s.valor into valor_hora
from salario_hora s
where s.cuil_empleado=cuil_guia
and s.fecha_desde=(
    select max(sal.fecha_desde)
    from salario_hora sal
    where sal.cuil_empleado=cuil_guia
    and fecha_desde ⩽ fecha_valor
);

RETURN valor_hora;
END$$

DELIMITER $$
USE `role_play_events`$$
CREATE PROCEDURE `costo_tours` (limite datetime)
BEGIN
select t.nro, t.tematica, t.cuil_guia, t.fecha_hora_salida,
t.fecha_hora_regreso
    , costo_hora_empleado(t.cuil_guia, t.fecha_hora_salida)
        *
timestampdiff(hour,t.fecha_hora_salida,t.fecha_hora_regreso)
costo_guia
    , sum(costo_hora_empleado(e.cuil_encargado,e.fecha_hora_ini)
        * timestampdiff(hour,e.fecha_hora_ini,e.fecha_hora_fin))
costo_encargados
    , (costo_hora_empleado(t.cuil_guia, t.fecha_hora_salida) *
timestampdiff(hour,t.fecha_hora_salida,t.fecha_hora_regreso))
        +
(sum(costo_hora_empleado(e.cuil_encargado,e.fecha_hora_ini) *
timestampdiff(hour,e.fecha_hora_ini,e.fecha_hora_fin))) costo_total
from tour t
inner join escala e
    on t.nro=e.nro_tour
where t.fecha_hora_regreso ⩽ limite
group by t.nro, t.tematica, t.cuil_guia, t.fecha_hora_salida,
t.fecha_hora_regreso
order by costo_total desc;
END$$

DELIMITER ;
```

```
call costo_tours(now());
```

Ej 3

Enunciado

****Salarios a categoría****La empresa decidió cambiar los salarios de los empleados de ser por empleado a por categoría y también deberá llevarse un registro histórico. Se debe:
a) Crear una entidad categoría, la misma debe identificarse por un código autoincremental y tener una descripción.
b) Crear una tabla salario_hora_categoria débil de categoría con una fecha desde y un valor.
c) Migrar los datos de las categorías de los empleados a la descripción de categoría y reemplazar la descripción actual con una clave foránea.
d) Cargar los salario_hora_categoria de cada categoría desde hoy, con el máximo salario actual de los empleados de dicha categoría
e) Limpiar los datos que ya no son necesarios.

Resolución

```mysql
CREATE TABLE `categoria` (
  `codigo` int unsigned NOT NULL AUTO_INCREMENT,
  `descripcion` varchar(255) NOT NULL,
  PRIMARY KEY (`codigo`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `salario_categoria_hora` (
  `codigo_categoria` int unsigned NOT NULL,
  `fecha_desde` date NOT NULL,
  `valor` decimal(10,3) NOT NULL,
  PRIMARY KEY (`codigo_categoria`,`fecha_desde`),
  CONSTRAINT `fk_salario_categoria_hora_categoria` FOREIGN KEY
(`codigo_categoria`) REFERENCES `categoria` (`codigo`) ON DELETE
RESTRICT ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

alter table empleado
add column codigo_categoria int unsigned null;
```

```
begin;

insert into categoria(descripcion)
select distinct categoria
from empleado;

update empleado e
inner join categoria c
    on e.categoria=c.descripcion
set e.codigo_categoria=c.codigo;

insert into salario_categoria_hora
with fec_val as (
    select sal.cuil_empleado, max(sal.fecha_desde) ult_fec
    from salario_hora sal
    where sal.fecha_desde ⩽ now()
    group by sal.cuil_empleado
)
select e.codigo_categoria, now(), max(sh.valor)
from salario_hora sh
inner join fec_val
    on sh.cuil_empleado=fec_val.cuil_empleado
    and sh.fecha_desde=fec_val.ult_fec
inner join empleado e
    on sh.cuil_empleado=e.cuil
group by e.codigo_categoria;

commit;

alter table empleado drop column categoria;
drop table salario_hora;

```