**01**

CREATE VIEW BungalowLFB AS

SELECT idBungalow, nomBungalow, superficieBungalow

FROM Bungalows b

JOIN Campings ca ON ca.idCamping = b.idCamping

WHERE nomCamping = ‘Les Flots Bleus’

SELECT COUNT(\*)

FROM BungalowLFB

**02**

CREATE VIEW LocationLFB AS

SELECT idLocation, c.idClient, nomClient, prenomClient, b.idBungalow, nomBungalow

FROM Location l

JOIN Clients c ON l.idClient = c.idClient

JOIN BungalowsLFB b on b.idBungalow = l.idBungalow

SELECT idBungalow, nomBungalow, COUNT(idLocation)

FROM LocationsLFB

GROUP BY idBungalow, nomBungalow

**03**

CREATE OR REPLACE VIEW LocationBungalows AS

SELECT b.idB ungalow, nomBungalow, b.idCamping, nomCamping, superficieBungalow,

COUNT(idLocation) AS nbLocations, NVL(SUM(montantLocation), 0) AS CA,

SUM((montantLocation)/COUNT(idLocation) AS CAML

FROM Bungalows b

LEFT JOIN Locations l ON b.idBungalow = l.idBungalow

JOIN Campings c ON b.idCamping = c.idCamping

GROUP BY b.idBungalow, nomBungalow, b.idCamping, nomCamping,

superficieBungalow

ORDER BY idCamping, idBungalow;

**04**

CREATE OR REPLACE VIEW ServicesBungalows AS

SELECT b.idBungalow, nomBungalow, b.idCamping, nomCamping, superficieBungalow,

COUNT(p.idService) AS nbServices, COUNT(DISTINCT categorieService) AS nbCategoriesService

FROM Bungalows b

LEFT JOIN Proposer p ON b.idBungalow = p.idBungalow

LEFT JOIN Services s ON p.idService = s.idService

JOIN Campings c ON b.idCamping = c.idCamping

GROUP BY b.idBungalow, nomBungalow, b.idCamping, nomCamping, superficieBungalow

ORDER BY idCamping, idBungalow;

**05**

CREATE OR REPLACE VIEW DonneesBungalow AS

SELECT lb.idBungalow, lb.nomBungalow, lb.idCamping, lb.nomCamping, lb.superficieBungalow, nbLocations, CA, CAML, nbServices, nbCategoriesService

FROM LocationsBungalow lb

JOIN ServicesBungalows sb ON lb.idBungalow = sb.idBungalow

ORDER BY idCamping, idBungalow;

SELECT nomBungalow, superficieBungalow

FROM DonneesBungalows

WHERE nbLocations = 0 AND nbServices > 2;

SELECT nomBungalow

FROM DonneesBungalows

WHERE nbLocations = (SELECT MAX(nbLocations)

FROM DonneesBungalows);

**08**

CREATE OR REPLACE VIEW EmployesSansCamping (idEmploye, nomEmploye, prenomEmploye, salaireEmploye, idEmployeChef) AS

SELECT idEmploye, nomEmploye, prenomEmploye, salaireEmploye, idEmployeChef

FROM Employes

WHERE idCamping IS NULL

A travers de la vue il est possible de:

* Insérer des lignes dans la table Employés.
* Modifier/Supprimer des lignes de la table Employés.

Pour savoir si les réponses aux questions précédentes sont exactes, essayer:

* D’insérer un nouvel employé dans la table *Employés* à travers la vue. *Par exemple, l’employé Judas Stiko, qui a l’identifiant E100, qui a un salaire de 3000€ et qui n’a pas de chef.*
* De modifier un employé de la table *Employés* à travers la vue. *Par exemple, changer le nom (par exemple Nana) de l’employé dont l’identifiant est E100.*
* De supprimer un employé de la table *Employés* à travers la vue. *Par exemple, l’employé dont l’identifiant est E100*.

**Tout fonctionne ✅**

**CREATE VIEW EmployeesSansCamping AS**

**SELECT idEmploye, nomEmploye, prenomEmploye, salaireEmploye, idEmployeChef**

**FROM Employes**

**WHERE idCamping IS NULL;**

**INSERT INTO EmployesSansCamping**

**VALUES(‘E100’, ‘Stiko’, ‘Judas’, 3000, NULL);**

**UPDATE EmployesSansCamping**

**SET nomEmploye = ‘Nana’**

**WHERE idEmploye = ‘E100’;**

**DELETE FROM EmployesSansCamping**

**WHERE idEmploye = ‘E100’;**

**09**

CREATE VIEW EMPLOYESAVECCAMPING (nomEmploye, prenomEmploye, salaireEmploye) AS

SELECT nomEmploye, prenomEmploye, salaireEmploye

FROM employes

WHERE idCamping IS NOT NULL;

A travers de la vue il est possible de:

* ~~Insérer des lignes dans la table Employés.~~
* Modifier/Supprimer des lignes de la table Employés.

Pour savoir si les réponses aux questions précédentes sont exactes, essayer:

* ~~Insérer un nouvel employé dans la table~~ *~~Employés~~* ~~à travers la vue.~~ *~~Par exemple, l’employé Judas Nana, qui a l’identifiant E100, qui a un salaire de 5000€et qui n’a pas de chef.~~*
* De modifier un employé de la table *Employés* à travers la vue. *Par exemple, changer le nom (par exemple Javel) de l’employé dont l’identifiant est E100.*
* De supprimer un employé de la table *Employés* à travers la vue. *Par exemple, l’employé dont le prénom est Aude*.

**CREATE OR REPLACE VIEW EmployesAvecCamping AS**

**SELECT nomEmploye, prenomEmploye, salaireEmploye**

**FROM Employes**

**WHERE idCamping IS NOT NULL;**

**/\* Impossible car vue sans identifiant \*/**

**INSERT INTO EmployesAvecCamping**

**VALUES(‘E200’, ‘Nana’, ‘Judas’, 5000);**

**UPDATE EmployesAvecCamping**

**SET nomEmploye = ‘Javel’**

**WHERE prenomEmploye = ‘Aude’;**

**DELETE FROM EmployesAvecCampoing**

**WHERE prenomEmploye = ‘Aude’;**

**10**

CREATE VIEW ClientsParVille(ville, nbClients) AS

SELECT villeClient, COUNT(DISTINCT idClient)

FROM clients

GROUP BY villeClient

A travers de la vue il est possible de:

* ~~Insérer des lignes dans la table Employés.~~
* ~~Modifier/Supprimer des lignes de la table Employés.~~

Pour savoir si les réponses aux questions précédentes sont exactes, essayer:

* ~~Insérer une nouvelle ville à travers la vue.~~ *~~Par exemple, la ville ‘Rodez’ dans laquelle on aurait 3 habitants~~*
* ~~De modifier une ville à travers la vue.~~ *~~Par exemple, changer le nom de la ville ‘Montpellier’~~*
* ~~De supprimer une ville à travers la vue.~~ *~~Par exemple, supprimer les villes pour lesquelles on a deux clients.~~*

**CREATE OR REPLACE VIEW ClientsParVille AS**

**SELECT villeClient AS ‘Ville’, COUNT(idClient) AS ‘nbClients’**

**FROM Clients**

**GROUP BY villeClient;**

**/\* impossible car GROUP BY \*/**

**INSERT INTO ClientsParVille**

**VALUES(‘Rodez’, 3);**

**/\* impossible car GROUP BY \*/**

**UPDATE ClientsParVille**

**SET villeClient = ‘Montpellier’**

**WHERE villeClient = ‘M’;**

**/\* impossible car GROUP BY \*/**

**DELETE FROM ClientsParVille**

**WHERE nbClients = 2;**

**11**

CREATE VIEW BungalowsEtCampings(idBungalow, nomBungalow, superficieBungalow, idCamping, nomCamping) AS

SELECT idBungalow, nomBungalow, superficieBungalow, b.idCamping, nomCamping

FROM bungalows b LEFT OUTER JOIN campings c

ON b.idCamping=c.idCamping

A travers de la vue il est possible de:

* ~~Insérer des lignes simultanément dans les tables~~ *~~Bungalows~~* ~~et~~ *~~Campings~~* ~~?~~
* Insérer des lignes dans la table *Bungalows* ?
* Insérer des lignes dans la table *Campings* ?
* Modifier/Supprimer des lignes de la table *Bungalows* ?
* Modifier/Supprimer des lignes de la table Campings ?

Pour savoir si les réponses aux questions précédentes sont exactes, essayer:

* ~~Insérer en même temps un nouveau bungalow et un nouveau camping à travers la vue.~~ *~~Par exemple, le bungalow B13, ‘Le Souterrain’ de 75m² qui se trouve dans le camping CAMP10 ‘Yellow Shark’.~~*
* Insérer uniquement un nouveau bungalow à travers la vue. *Par exemple, le bungalow B14 ‘Le Dépotoir’ de 25m².* Si l’insertion n’est pas possible, essayer de renseigner le nom des colonnes (et uniquement celles-là) dans l’INSERT.
* Modifier un bungalow de la table *Bungalows* à travers la vue. *Par exemple, changer la superficie du bungalow ‘Le Palace’*.
* Modifier un camping de la table *Campings* à travers la vue. *Par exemple, changer le nom du camping ‘The White Majestic’ et l’appeler ‘Le Majestique Blanc’*.

**CREATE VIEW BungalowsEtCampings AS**

**SELECT b.inBungalow, nomBungalow, superficieBungalow, ca.idCamping, nomCamping**

**FROM Bungalows b**

**JOIN Campings ca ON ca.idCamping = b.idCamping;**

**/\* impossible car multi-tables \*/**

**INSERT INTO BungalowsEtCampings**

**VALUES(‘B11’, ‘Le Souverain’, 70, ‘CAMP1’, ‘Les Flots Bleus’);**

**/\* possible modifier table Bungalows car clé préservée \*/**

**UPDATE BungalowsEtCampings**

**SET superficieBungalow = 30 WHERE idBungalow = ‘B1’;**

**/\* impossible modifier table campings car clé non préservée \*/**

**UPDAtE BungalowsEtCampings**

**SET nomCamping = ‘toto’ WHERE idCamping = ‘CAMP1’;**

**12**

CREATE VIEW CampingsPalavas(idCamping, nomCamping, villeCamping, nbEtoilesCamping) AS

SELECT idCamping, nomCamping, villeCamping, nbEtoilesCamping

FROM campings

WHERE villeCamping='Palavas'

A travers de la vue il est possible de:

* Insérer dans la table *Campings* des campings qui se trouvent dans la ville de Palavas ?
* Insérer dans ta table *Campings* des campings qui ne se trouvent pas dans la ville de Palavas ?

Pour savoir si les réponses aux questions précédentes sont exactes, essayer:

* Insérer, à travers la vue, un nouveau camping qui se trouve dans la ville de Carnon. *Par exemple, le camping CAMP4, qui se nomme ‘El Delfin Azul’ qui se trouve à Carnon et possède 3 étoiles.*Vérifiez ensuite s’il est possible de voir ce nouveau camping à travers la vue. Puis vérifier si le nouveau camping a été inséré ou non dans la table *Campings*.
* Modifier la vue *CampingPalavas* (CREATE OR REPLACE VIEW) afin qu’à travers cette vue, on ne puisse insérer que des campings qui se trouvent dans la ville de Palavas.
* Modifier à nouveau la vue pour qu’elle soit à présent "non modifiable" ; c'est-à-dire qu’il ne soit plus possible d’insérer, de modifier ou de supprimer des données à travers la vue.

CREATE VIEW CAMPINGSPALAVAS AS

SELECT idCamping, nomCamping, villeCamping, nbEtoilesCamping

FROM Campings

WHERE villeCamping = ‘Palavas’;

INSERT INTO CampingsPalavas

VALUES (‘CAMP4’, ‘E1 Delfin Azul’, ‘CARNON’, 3);

CREATE OR REPLACE VIEW CAMPINGSPALAVAS AS

SELECT idCamping, nomCamping, villeCamping, nbEtoilesCamping

FROM Campings

WHERE villeCamping = ‘Palavas’

WITH CHECK OPTION;

CREATE OR REPLACE VIEW CAMPINGSPALAVAS AS

SELECT idCamping, nomCamping, villeCamping, nbEtoilesCamping

FROM Campings

WHERE villeCamping = ‘Palavas’

WITH READ ONLY;