/\*\*\*\*\*\*\*\*EXEMPLE\*\*\*\*\*\*\*\*\*\*\*/

DELIMITER $$

/\* Procédure permettant l'affichage de tous les utilisateurs selon

leur rôle \*/

CREATE PROCEDURE afficheUtilisateursSelonRole(In nomRole VARCHAR(45))

BEGIN

DECLARE idRole INT;

/\* On récupère l'identifiant du rôle à partir de son nom \*/

SELECT( idRoleParNomRole(nomRole) ) INTO @idRole;

SELECT \* FROM Utilisateurs

WHERE idUtilisateur IN ( SELECT Utilisateurs\_idUtilisateur FROM Possede WHERE Roles\_idRole = @idRole );

END $$

DELIMITER ;

/\*\*\*\*\*\*\* afficheStatutCandidat(idCandidat)\*/ manque des données dans hist-statuts

DELIMITER $$

CREATE PROCEDURE afficheStatutCandidat(In candidat INT)

BEGIN

declare maxdate DATETIME;

IF ( candidat IN ( SELECT fk\_idCandidat FROM Candidats) ) THEN

set @maxdate = (select max(date) date from Hist\_Statuts

where fk\_idCandidat=candidat);

select statut from Hist\_Statuts

where date=maxdate and fk\_idCandidat=candidat;

ELSE

CALL raiseError("Ce n'est pas un candidat.");

END IF;

END $$

DELIMITER ;

/\*\*\*\*\*\*\* afficheStatutDossier(idCandidat)\*/ Validée

DELIMITER $$

CREATE PROCEDURE afficheStatutDossier(In candidat INT)

BEGIN

IF ( candidat IN ( SELECT fk\_idCandidat FROM Candidats) ) THEN

(select CV, LM, CNI, releve1, releve2 from Candidats where fk\_idCandidat=candidat);

ELSE

CALL raiseError("Ce n'est pas un candidat.");

END IF;

END $$

DELIMITER ;

/\*\*\*\*\*\*\* afficheHistStatutsCandidat(idCandidat)\*/manque de données

DELIMITER $$

CREATE PROCEDURE afficheHistStatutsCandidat(In candidat INT)

BEGIN

IF ( candidat IN ( SELECT fk\_idCandidat FROM Candidats) ) THEN

select statut, date from Hist\_Statuts where fk\_idCandidat=candidat;

ELSE

CALL raiseError("Ce n'est pas un candidat.");

END IF;

END $$

DELIMITER ;

/\*\*\*\*\*\*\* afficheEntretienCandidat(idCandidat)\*/ manque de données mais syntaxe verifiée

DELIMITER $$

CREATE PROCEDURE afficheEntretienCandidat(In candidat INT)

BEGIN

IF ( candidat IN ( SELECT fk\_idCandidat FROM Candidats) ) THEN

select dateDebut, dateFin salle, Jury\_idJury from Entretiens where Candidats\_idCandidat=candidat;

ELSE

CALL raiseError("Ce n'est pas un candidat.");

END IF;

END $$

DELIMITER ;

/\*\*\*\*\*\*\* afficheCourrielCandidatsStatut(idCandidat)\*/

DELIMITER $$

CREATE PROCEDURE afficheCourrielCandidatsStatut(In s ENUM('inscrit', 'admissible', 'admis', 'integre', 'refuse'))

BEGIN

if(s!='refuse') then

select u.courriel from utilisateurs u, (select hs.date as date, hs.statut, hs.fk\_idCandidat as candidat from Hist\_Statuts hs where hs.statut=s) statut where u.idUtilisateur=statut.candidat;

else

select 'Essayer afficheCourrielRefuse1() ou afficheCourrielRefuser2().';

end if;

END $$

DELIMITER ;

/\*\*\*\*\*\*\* afficheCourrielRefuse2()\*/

DELIMITER $$

CREATE PROCEDURE afficheCourrielRefuse2()

BEGIN

select u.courriel

from utilisateurs u,

( select hs.statut, hs.fk\_idCandidat as candidat

from Hist\_Statuts hs

where hs.statut='refuse'

and hs.fk\_idCandidat in

( select hst.fk\_idCandidat

from Hist\_Statuts hst

where statut='admissible' ) ) statut

where u.idUtilisateur=statut.candidat;

END $$

DELIMITER ;

/\*\*\*\*\*\*\* afficheCourrielRefuse1()\*/

DELIMITER $$

CREATE PROCEDURE afficheCourrielRefuse1()

BEGIN

select u.courriel

from utilisateurs u,

( select hs.statut, hs.fk\_idCandidat as candidat

from Hist\_Statuts hs

where hs.statut='refuse'

and hs.fk\_idCandidat not in

( select hst.fk\_idCandidat

from Hist\_Statuts hst

where statut='admissible' ) ) statut

where u.idUtilisateur=statut.candidat;

END $$

DELIMITER ;

/\*\*\*\*\*\*\* afficheCandidatsStatut(statut)\*/

DELIMITER $$

CREATE PROCEDURE afficheCandidatsStatut(In s ENUM('inscrit', 'admissible', 'admis', 'integre', 'refuse'))

BEGIN

select u.nom, u.prenom

from utilisateurs u, candidats c, hist\_statuts hs

where u.idUtilisateur=c.fk\_idCandidat

and hs.statut=s;

END $$

DELIMITER ;

/\*\*\*\*\*\*\*afficheHistStatutCandidat(idCandidat) \*/

DELIMITER $$

CREATE PROCEDURE afficheHistStatutCandidat(candidat INT)

BEGIN

IF ( candidat IN ( SELECT fk\_idCandidat FROM Candidats) ) THEN

select hs.statut, hs.date

from Hist\_Statuts hs

where hs.fk\_idCandidat=candidat

order by date ASC;

else

CALL raiseError("Ce n'est pas un candidat.");

end if;

END $$

DELIMITER ;

/\*\*\*\*\*\*\*afficheCandidatsDossierComplet() \*/

DELIMITER $$

CREATE PROCEDURE afficheCandidatsDossierComplet()

BEGIN

select nom, prenom

from candidats c, utilisateurs u

where c.fk\_idCandidat=u.idUtilisateur

and cv=1 and lm=1 and cni=1 and releve1=1 and releve2=1;

END $$

DELIMITER ;

/\*\*\*\*\*\*\*afficheCandidatsDossierIncomplet() \*/

DELIMITER $$

CREATE PROCEDURE afficheCandidatsDossierIncomplet()

BEGIN

select nom, prenom

from candidats c, utilisateurs u

where c.fk\_idCandidat=u.idUtilisateur

and c.fk\_idCandidat not in

(select c2.fk\_idCandidat

from Candidats c2

where c2.cv=1

and c2.lm=1

and c2.cni=1

and c2.releve1=1

and c2.releve2=1);

END $$

DELIMITER ;

/\*\*\*\*\*\*\*afficheidCandidat(nom, prenom) \*/

DELIMITER $$

CREATE PROCEDURE afficheidCandidat(nom varchar(45), prenom varchar(45))

BEGIN

select idUtilisateur from utilisateurs u where u.nom=nom and u.prenom=prenom;

END $$

DELIMITER ;

/\*\*\*\*\*\*\*afficheCandidatsNoteSuperieure(note)\*/

DELIMITER $$

CREATE PROCEDURE afficheCandidatsNoteSuperieure(note int)

BEGIN

if(note<20 and note>0) then

select u.nom, u.prenom, e.note

from utilisateurs u, entretiens e

where e.candidats\_idcandidat=u.idUtilisateur

group by e.note

having e.note>note;

else

call raiseError("La note choisie n'est pas valide.");

end if;

END $$

DELIMITER ;

/\*\*\*\*\*\*\*afficheNombreCandidatsNoteSuperieure(note)\*/

DELIMITER $$

CREATE PROCEDURE afficheNombreCandidatsNoteSuperieure(note int)

BEGIN

if(note<20 and note>0) then

select count(e.Candidats\_idCandidat) from entretiens e where e.note>note;

else

call raiseError("La note choisie n'est pas valide.");

end if;

END $$

DELIMITER ;