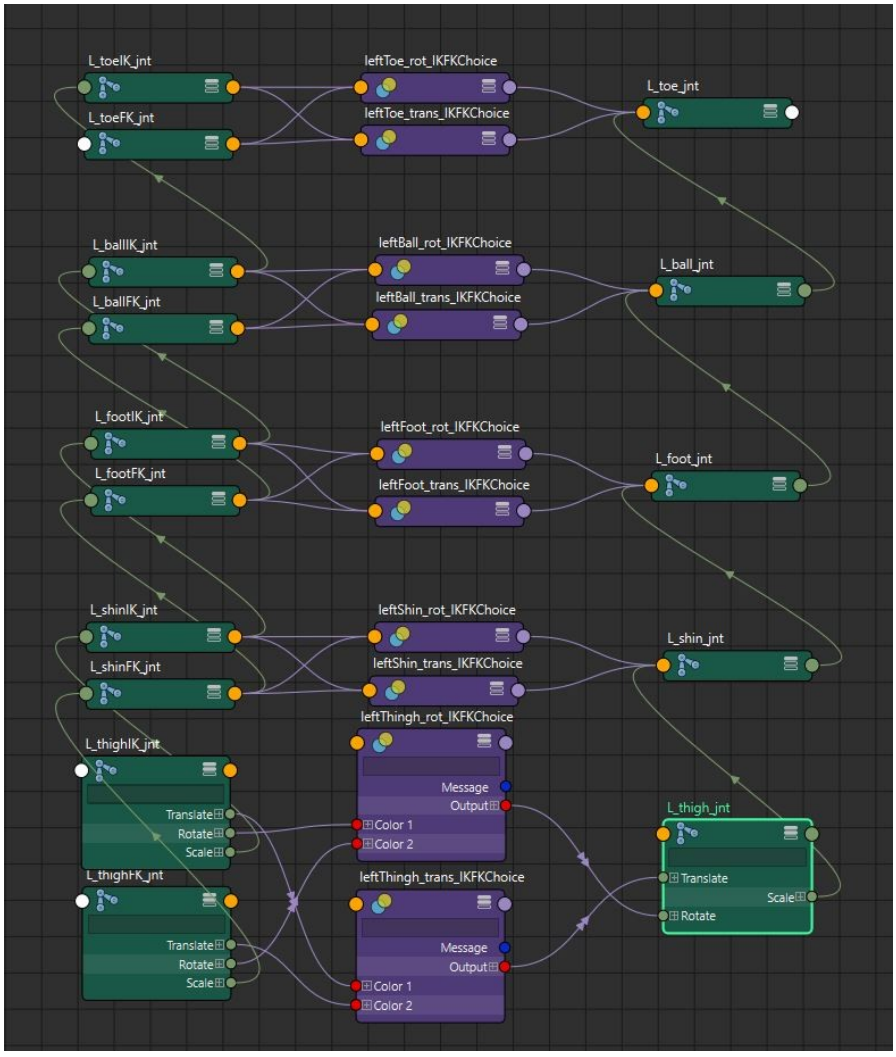
LEG IK / FK & FOOT ROLL

- **Joint Setup:**

  - Creo una catena di 5 Joint che rinomino in **L_thigh_jnt**, **L_shin_jnt**, **L_foot_jnt**, **L_ball_jnt** e **L_toe_jnt**.
  - Col Multicut taglio la mesh all'altezza dei joint e rinomino le mesh proxy.
  - Oriento la catena di joint con l'asse primario X, quello secondario Y e il secondario Wolrd Orientation X positive, in modo che la X guidi e la Y punti verso l'esterno.
  - Duplico la catena due volte e rinomino una delle due catene con IK e l'altra con FK nei nomi dei joint (es.: **L_thighFK_jnt**).

- **IK FK Switch:**

  Per creare lo switch tra IK e FK utilizzerò un sistema a nodi anziché una serie di constraints. Principalmente perché i nodi hanno una velocità di calcolo maggiore e consumano meno risorse, e inoltre ci permettono di animare eventualmente uno switch progressivo.
  - Apro il node editor e chiamo dentro tutte e tre le catene di joint.
  - Creo 5 diversi nodi colorBlend ognuno chiamato **left_"nome parte della gamba"_rot_IKFKChoice**.
  - In ognuno di loro inserisco come color 1 il valore rotate del joint IK e nel color 2 il valore rotate del joint FK al quale il nodo fa riferimento.
  - L'output lo inserisco nel rotate della catena joint principale.
  - Creo altri 5 colorBlend, rinomino **left_"nome parte della gamba"_trans_IKFKChoice**, e ripeto il procedimento precedente ma applicandolo ai translate.



- Creo un controllo tramite il “Create → Type” e scriverò “IK / FK”
- Selezione la mesh risultante e vado su “Modify → Convert → Type to Curve” e cancello la mesh vecchia.
- Unisco le lettere ed eventuali altre curve d’estetica col comando “parent -r -s” e posiziono la curva risultante dietro il piede.
- Rinomino in **L_legIKFKSwitch_ctrl**, la gruppo e rinomino il gruppo **L_legCtrl_grp**.
- Selezionando la curva aggiungo un nuovo attributo (Channel Box → Edit → Add Attribute), lo chiamo FK / IK Blend e lo metto al minimo di 0 e massimo di 1

Dal momento che questo attributo ha valori tra 0 e 1 e dobbiamo usarlo per controllare i nodi blendColor, i quali anch’essi vanno da 0 a 1, possiamo limitarci ad usare il Connection Editor.

- Apro il Connection Editor (Windows → General Editors → Connection Editor).
- A sinistra carico il controllo switch.
- Per caricare a destra i nodi blendColor vado prima sull’outliner e sotto Display tolgo la spunta da “DAG Objects Only”, cerco i nodi, li seleziono tutti e 10 insieme e li carico a destra.
- Collego l’attributo nuovo della curva di controllo ad ogni “blender” di ognuno dei 10 nodi.

- **FK Controls & Stretch:**

Alla nostra catena FK andremo a creare i nuovi controlli ma vogliamo aggiungere anche un modo per poter controllare manualmente lo stretch della coscia e dello stinco.

- Creo 4 curve, le posiziono sulla coscia, stinco, piede e dita, rinomino in **L_thighFK_ctrl**, **L_shinFK_ctrl**, **L_footFK_ctrl** e **L_ballFK_ctrl**. Freeze trasformazioni e cancello le storie.
- Setto il rotate order dei controlli appena creati in “XZY”.
- Questo stesso rotate order lo imposto anche su ogni joint di tutte e tre le catene.
- Do un orient constraint dalle curve ai rispettivi joint FK e imparento (P) i controlli nello stesso ordine dei joint.

Ho dato un orient anziché il solito parent constraint perché ora voglio sfruttare il translate X delle ossa per determinare lo squash/stretch della caviglia e della coscia.

- Creo un nuovo attributo sia sulla coscia che sullo stinco chiamato **Lenght** con minimo 0, nessun massimo e default a 1

- Uso il Set Driven Key per caricare come driver il controllo della coscia e come driven l’osso dello stinco.
- Metto una chiave tra l’attributo Lenght e Translate X così come sono ora → imposto a 0 sia il Lenght e il Translate X e metto una seconda chiave.
- Apro il Graph Editor (Windows → Animation Editors → Graph Editors), seleziono l’osso dello stinco, seleziono la sua curva e faccio click destro → Tangents → Linear.
- Sotto “Curves” clicco su post infinity → Cycle with Offset, per far continuare la curva all’infinito e per, in questo modo, poter stretchare l’arto all’infinito.

- Ripeto il procedimento ma utilizzando il controllo dello stinco col suo Lenght e l’osso del foot col suo Translate X.

- Do un Point Constraint tra l'osso dello stinco e il suo controllo e l'osso del piede col suo controllo, in modo che quando l'arto si stretcha i controlli seguono la posizione del joint.

- Dal momento che la mesh non si stretcha insieme ai joint userò il valore di Lenght per guidare lo scale X delle mesh proxy, e ciò posso farlo tramite il Connection Editor.
Se le mesh proxy sono orientate con le ossa mi basterà creare la connessione, ma se ciò non dovesse essere il caso allora procedo a gruppare prima la mesh in un gruppo orient che manualmente ruoto per far coincidere il suo orientamento con quello dei joint.

- **IK handle setup:**

- Creo l'IK tramite Skeleton → IK Handle → Rotate-Plane Solver e lo imposto dal joint della coscia al joint del piede.
- Rinomino lo handle **L_leg_eff.**

- Creo una nuova curva e la posiziono sotto il piede ed uso i suoi vertici per darle una forma adatta al caso.
- Setto il pivot al joint del piede e freezo/cancello storie.
- Imposto l'ordine di rotazione in ZXY

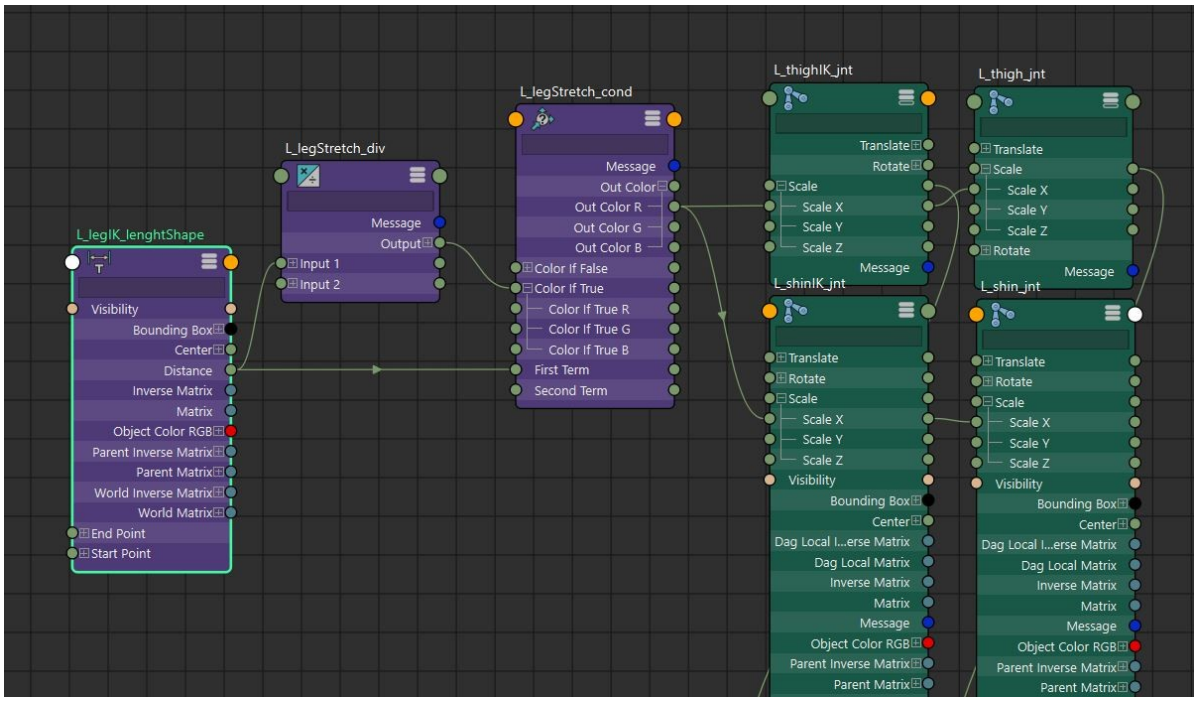
- Imparento (P) l'IK handle sotto il controllo del piede.
- Creo due nuovi IK Handle, tra il joint foot e il joint ball; e tra il joint ball e il joint toe.
- Rinomino gli handle in **L_ball_hdl** e **L_toe_hdl**.
- Imparento anche loro due dentro il controllo del piede.

- **Squash/Stretch controlli IK:**

Per fare in modo che la gamba si stretchi in modalità IK non posso usare l'info node usato per la spine, perché diversamente da allora non abbiamo una curva. Useremo questa volta un altro oggetto, ovvero il Measure Tool per avere un qualcosa su cui calcolare la lunghezza dell'arto.

- Creo due locator di misurazione tramite Ctreate → Measure Tools → Distance Tool, e li posiziono sul joint thigh e sul joint foot.
- Rinomino i locator **L_legIK_leghtStart_loc** e **L_legIK_leghtEnd_loc**, e la misura in **L_legIK_lenght**.

- Apro il Node Editor e chiamo dentro L_legIK_lenght e entrambi i joint della coscia e caviglia, sia della catena IK che quella originale. Voglio tenere solo **L_legIK_lenghtShape**, quindi L_legIK_lenght posso rimuoverlo dall'editor.
- Creo un nodo "multiplyDivide" e lo rinomino **L_legStretch_div** e ci metto come input 1 il "distance" di L_legIK_lenghtShape, mentre nell'input2 ci copio-incollo lo stesso valore e setto l'operazione in una divisione dall'Attribute Editor.
- Creo un nodo "condition" e lo rinomino in **L_legStretch_cond** e nel suo input "Color if True" inserisco l'output del nodo precedente, mentre nel "First Term" inserisco il distance dello shape di L_legIK_lenghtShape. Il second term sarà di nuovo copia-incollato del suo valore a riposo e l'operazione la metto in "Greater or Equal".
- L'output della condizione lo metto nello scale X sia della coscia che dello stinco (IK).
- L'output scale X degli stessi nodi li metto nell'input Scale X dei corrispettivi ossi della catena originale.



- **Pole Vector:**

  - Rimanendo in modalità IK creo una curva NURBS e la posiziono frontale al ginocchio.
  - Freezo / cancello le storie e rinomino in **L_kneePV_ctrl**.
  - Seleziono il controllo, poi lo handle dell'IK della gamba e sotto Skeleton seleziono il Pole Vector.

  - Creo una curva col CV Curve Tool con due soli vertici e la rinomino **L_PVconnector_crv**.
  - Sotto Deform, cerco Cluster e ne applico uno per vertice della curva.
  - Snappo uno dei vertici sul joint del ginocchio e l'altro lo snappo sulla curva di controllo del Pole Vector.
  - Do un Point Constraint col mantain offset sui cluster per fare in modo che la curva si deformi per connettere visivamente il controllo del Pole Vector e il ginocchio.

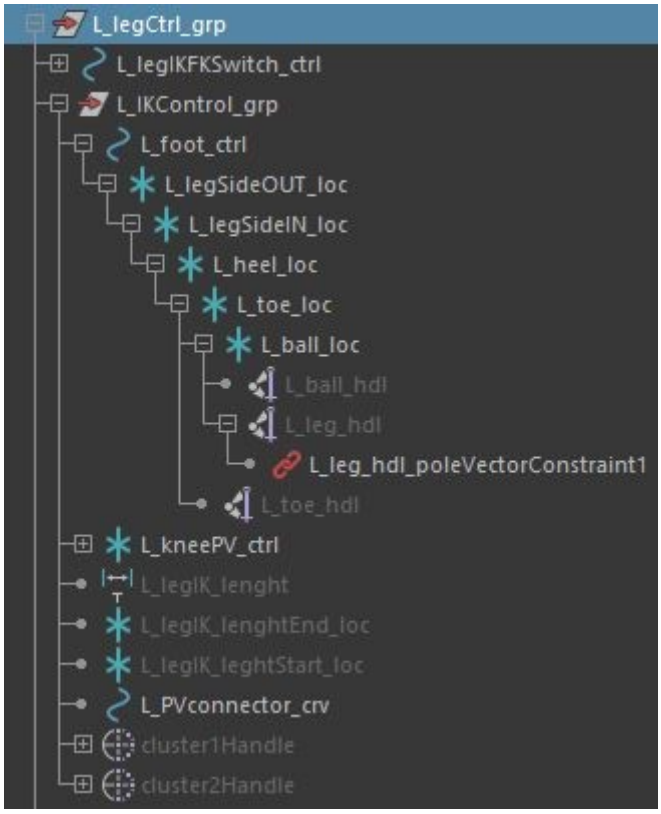
  - Do un Aim Constraint dal joint del ginocchio IK al controllo NURBS facendo bene attenzione all'asse aim da impostare.

- **Foot Roll:**

  - Creo 5 nuovi Locator e li rinomino **L_heel_loc**, **L_toe_loc**, **L_ball_loc**, **L_legSideIN_loc** e **L_legSideOUT_loc**.
  - Li posiziono rispettivamente sulla base del tallone, sui loro joint con lo stesso nome, mentre i sideIN e sideOUT rispettivamente a lato interno ed esterno del piede.

  - Imparento L_leg_hdl e L_ball_hdl dentro L_ball_loc.
  - Imparento L_ball_loc dentro L_toe_loc.
  - Imparento L_toe_hdl dentro L_toe_loc.
  - Imparento L_toe_loc dentro L_heel_loc.
  - Imparento L_heel_loc dentro L_legSideIN_loc.
  - Imparento L_legSideIN_loc dentro L_legSideOUT_loc.

La gerarchia dovrebbe essere così:



- Seleziono la curva di controllo IK del piede e creo 6 nuovi Attributi: **HeelRoll**, **BallRoll**, **ToeRoll**, **HeelPivot**, **ToePivot**, **SideToSide**.
- Tutti avranno un default di 0 e nessun limite minimo e massimo.

- Provvedo ad usare il Connection Editor per connettere gli attributi nuovi con i giusti valori di rotazione dei Locator, facendo attenzione a limitare le informazioni di rotazione sui locator L_legSideOUT_loc e L_legSideIN_loc, per fare in modo che si alternino correttamente.