**Aplicando regras no fluxo do protocolo**

Arquivos:

* 01-criacao-tabelas-REFORMADO.sql
* 02-dados-iniciais-REFORMADO.sql
* 03-engine-exemplo-uso.sql

**O que mudou (resumo)**

* mantive suas tabelas-base (setores, usuários, tipos, fluxos, protocolos, histórico) e garanti colunas necessárias (ValorTotal, EtapaAtual, SetorAtualId, Status).
* adicionei o **módulo de regras** (Regras, RegraCondicoes, RegraAcoes, ProtocoloMetadados) e a tabela de **EtapasExtras** para inserir aprovações **entre ordens**.
* criei a **engine**:
  + fn\_RegraSatisfeita(@RegraId,@ProtocoloId) — checa se a regra bate (suporta ValorTotal e metadados =/LIKE).
  + sp\_AplicarRegras\_Protocolo(@ProtocoloId) — retorna a primeira ação aplicável (prioridade mais alta).
  + sp\_TratarAcaoRegra(...) — executa a ação (ex.: INSERIR\_ETAPA\_INTERMEDIARIA, PULAR\_PARA\_ORDEM) e loga no histórico.
  + sp\_ProximaParada(@ProtocoloId) — calcula a próxima “parada” considerando **EtapasExtras pendentes** antes da próxima ordem e, se não tiver, segue o fluxo padrão por faixa de valor.
  + vw\_Protocolo\_EtapasExtrasPendentes — visão para debugar.

**Semente pronta (dados iniciais)**

o arquivo 02 cria um fluxo simples de **Nota Fiscal**:

* Ordem 1 → **Financeiro** (até 5.000)
* Ordem 2 → **Diretoria** (acima de 5.000)

e inclui a regra de “valor mais alto” (ex.: >= 20.000): **insere uma etapa intermediária antes da ordem 2** exigindo aprovação do **usuário Diretor**.

também criei três protocolos de exemplo:

* NF-0001 (R$ 3.000) → segue só a ordem 1.
* NF-0002 (R$ 12.000) → vai 1 → 2 normalmente.
* NF-0003 (R$ 25.000) → **dispara a etapa intermediária** entre 1 e 2.

**como usar no projeto (passo a passo)**

1. **criação/edição do protocolo**
   * ao criar (ou mudar ValorTotal/metadados), chame:
   * -- 1) avalia regras
   * SELECT \* FROM dbo.sp\_AplicarRegras\_Protocolo(@ProtocoloId);
   * -- 2) se retornar ação, execute:
   * EXEC dbo.sp\_TratarAcaoRegra @ProtocoloId, @AcaoTipo, @Parametro1, @Parametro2;
   * exemplo de ação que vem para valor alto de NF:  
     INSERIR\_ETAPA\_INTERMEDIARIA | ANTES\_DA\_ORDEM:2 | USUARIO:3
2. **descobrir a próxima etapa/destino** (na hora de encaminhar)
   * sempre consulte:
   * EXEC dbo.sp\_ProximaParada @ProtocoloId;
   * se existir **EtapaExtra pendente** para AntesDaOrdem = (EtapaAtual+1), o retorno será do tipo ETAPA\_EXTRA (com UsuarioId ou SetorId alvo).
   * se não existir, o retorno será do tipo FLUXO, apontando o SetorDestinoId configurado para aquela Ordem e faixa de valor.
3. **aprovação da etapa intermediária**
   * quando o usuário/ setor responsável aprovar, marque a etapa como concluída:
   * UPDATE dbo.ProtocoloDigital\_EtapasExtras
   * SET Status = 'Aprovado'
   * WHERE Id = @EtapaExtraId; -- veio no retorno da sp\_ProximaParada
   * registre no histórico (opcional) e chame sp\_ProximaParada novamente para seguir o curso normal.
4. **avanço da ordem**
   * quando você efetivamente mover para a próxima ordem, atualize:
   * UPDATE dbo.ProtocoloDigital\_Protocolos
   * SET EtapaAtual = ISNULL(EtapaAtual,0) + 1,
   * SetorAtualId = (SELECT TOP 1 SetorId FROM dbo.sp\_ProximaParada(@ProtocoloId) AS r WHERE r.Tipo='FLUXO');
   * (na prática, você vai usar o resultado de sp\_ProximaParada para preencher o destino certo).
5. **observabilidade**
   * qualquer ação de regra é logada em ProtocoloDigital\_Historico para trilha de auditoria.
   * debugue as extras com a view vw\_Protocolo\_EtapasExtrasPendentes.

**extensões fáceis**

* adicionar mais operadores/ campos na fn\_RegraSatisfeita (ex.: TipoProtocolo, SetorAtualId, data, etc.).
* criar ações adicionais (MUDAR\_SETOR\_DESTINO, INSERIR\_APROVADOR\_USUARIO) e tratá-las em sp\_TratarAcaoRegra.
* permitir múltiplas **EtapasExtras** (o modelo já suporta várias pendências antes da mesma ordem — você decide a política de execução, hoje pega a 1ª).