Logique paiements et écheanciers - Spécifications techniques

Application : dirCops
Date : 2026-02-09
Objectif : Définir la logique métier et technique pour les paiements et écheanciers, inspirée des bonnes pratiques ERP tout en restant simple.

---

1. Principes généraux

1.1 Source de vérité

- **Facture** : montant HT fixe (ne change pas après création).
- **Paiements** : liste des paiements enregistrés (un paiement = un enregistrement).
- **Encours** : calculé à la volée = `montant_ht - SUM(paiements.montant)`.
- **État facture** : dérivé de l'encours (`encours <= 0` → Payée, sinon → En attente).

1.2 Règles métier

1. **Un paiement est toujours positif** : `montant > 0`.
2. **Un paiement est lié à une facture** : `facture_id` obligatoire.
3. **L'encours ne peut pas être négatif** : si `SUM(paiements) > montant_ht`, l'encours = 0 (pas de crédit client géré).
4. **Un écheancier = un plan de paiement** : une facture peut avoir au plus un écheancier.
5. **Montant par échéance** : calculé au moment de la création de l'écheancier = `encours_actuel / nb_echeances`.

---

2. Calcul de l'encours

2.1 Formule standardisée

```
encours_facture = GREATEST(0, montant_ht - COALESCE(SUM(paiements.montant), 0))
```

Explication :

- `GREATEST(0, ...)` : garantit que l'encours ne soit jamais négatif.
- `COALESCE(..., 0)` : si aucun paiement, somme = 0.
- Arrondi à 2 décimales pour éviter les problèmes de précision.

2.2 Encours par client

```
encours_client = SUM(encours_facture) pour toutes les factures du client
```

2.3 Cas limites gérés

Cas	Comportement
Paiement supérieur au montant HT	Encours = 0 (pas de crédit)
Plusieurs paiements partiels	Encours = montant_ht - somme des paiements
Aucun paiement	Encours = montant_ht
Arrondis décimaux	Utilisation de ROUND() pour éviter les écarts

---

3. Écheancier de paiement

3.1 Structure

- **Une facture = un écheancier maximum** (contrainte d'unicité).
- **Champs** : `facture_id`, `nb_echeances`, `date_debut_echeancier`, `montant_par_echeance` (calculé et stocké).

3.2 Calcul du montant par échéance

Au moment de la création de l'écheancier :

```
montant_par_echeance = ROUND(encours_actuel / nb_echeances, 2)
```

Exemple :

- Facture : 1000 € HT
- Paiements déjà reçus : 200 €
- Encours actuel : 800 €
- Nombre d'échéances : 3
- Montant par échéance : 800 / 3 = 266.67 €

3.3 Dates d'échéances (calculées, non stockées)

Les dates sont calculées à l'affichage :

- Échéance 1 : `date_debut_echeancier`
- Échéance 2 : `date_debut_echeancier + 1 mois`
- Échéance 3 : `date_debut_echeancier + 2 mois`
- etc.

Fréquence : mensuelle (toujours).

3.4 Règles de création

1. **Un écheancier ne peut être créé que si** :
   - La facture existe.
   - La facture n'a pas déjà d'écheancier.
   - La facture a un encours > 0 (sinon pas besoin d'écheancier).
2. **Le montant par échéance est calculé au moment de la création** :
   - Basé sur l'encours actuel (montant_ht - paiements déjà reçus).
   - Stocké dans `echeanciers.montant_par_echeance` pour éviter de recalculer.

---

4. Base de données

4.1 Contraintes

Table	Contrainte	Détail
`paiements`	`montant > 0`	Un paiement doit être positif
`paiements`	`facture_id` FK	Référence facture obligatoire
`echeanciers`	`facture_id` UNIQUE	Une facture = un écheancier max
`echeanciers`	`nb_echeances > 0`	Au moins une échéance
`echeanciers`	`montant_par_echeance > 0`	Montant positif

4.2 Index recommandés

```
CREATE INDEX idx_paiements_facture_id ON paiements(facture_id);
CREATE INDEX idx_paiements_date ON paiements(date_paiement);
CREATE UNIQUE INDEX idx_echeanciers_facture_id ON echeanciers(facture_id);
```

4.3 Champs ajoutés (amélioration)

Table `echeanciers` :

- `montant_par_echeance DECIMAL(12,2)` : montant calculé au moment de la création.

---

5. Logique applicative

5.1 Enregistrement d'un paiement

Avant insertion :

1. Vérifier que la facture existe.
2. Vérifier que `montant > 0`.

3. Vérifier que `date_paiement` est valide.

Après insertion :

- L'encours est recalculé automatiquement lors des prochaines requêtes.
- Pas de mise à jour explicite nécessaire (calcul à la volée).

5.2 Création d'un échéancier

Avant insertion :

1. Vérifier que la facture existe.
2. Vérifier que la facture n'a pas déjà d'échéancier.
3. Calculer l'encours actuel de la facture.
4. Vérifier que `encours > 0`.
5. Calculer `montant_par_echeance = ROUND(encours / nb_echeances, 2)`.

Insertion :

```
INSERT INTO echeanciers (facture_id, nb_echeances, date_debut_echeancier, montant_par_echeance)
VALUES (?, ?, ?, ?)
```

5.3 Calcul de l'encours (fonction réutilisable)

En SQL :

```
SELECT
    f.id,
    f.montant_ht,
    COALESCE(SUM(p.montant), 0) AS total_paye,
    GREATEST(0, f.montant_ht - COALESCE(SUM(p.montant), 0)) AS encours_facture,
    CASE
        WHEN GREATEST(0, f.montant_ht - COALESCE(SUM(p.montant), 0)) <= 0 THEN 'paye'
        ELSE 'en_attente'
    END AS etat
FROM factures f
LEFT JOIN paiements p ON p.facture_id = f.id
WHERE f.id = ?
GROUP BY f.id, f.montant_ht
```

En PHP (mock) :

```php
function calculerEncoursFacture($factureId, $factures, $paiements) {
    $f = array_filter($factures, fn($x) => $x['id'] == $factureId)[0] ?? null;
    if (!$f) return null;

    $totalPaye = array_sum(array_map(
        fn($p) => (float)$p['montant'],
        array_filter($paiements, fn($p) => $p['facture_id'] == $factureId)
    ));

    $encours = max(0, (float)$f['montant_ht'] - $totalPaye);
    return [
        'montant_ht' => (float)$f['montant_ht'],
        'total_paye' => $totalPaye,
        'encours_facture' => $encours,
        'etat' => $encours <= 0 ? 'paye' : 'en_attente'
    ];
}
```

---

6. Cohérence des données

6.1 Vérifications à faire

Vérification	Quand	Action si échec
Facture existe	Avant paiement/écheancier	Erreur 404
Montant paiement > 0	Avant insertion paiement	Erreur 400
Encours > 0	Avant création écheancier	Erreur 400 "Facture déjà soldée"
Pas d'écheancier existant	Avant création écheancier	Erreur 400 "Écheancier déjà existant"

6.2 Normalisation des montants

- **Stockage** : `DECIMAL(12,2)` (2 décimales).
- **Calculs** : utiliser `ROUND()` pour éviter les écarts.
- **Comparaisons** : utiliser `<= 0.01` pour considérer comme nul (tolérance d'arrondi).

---

7. Échéances à venir (tableau de bord)

Les « échéances à venir » affichées sur le tableau de bord sont calculées à partir des **écheanciers** lorsque la facture en possède un, sinon à partir de la date de facture.

7.1 Règles de calcul

- **Facture avec écheancier :** pour chaque échéance prévue (i = 0 à nb_echeances − 1), on crée une ligne avec :
  - **date_echeance** = `date_debut_echeancier + i mois`
  - **montant_echeance** = `montant_par_echeance` (stocké dans l'écheancier)

- Seules les échéances dont la date est ≥ **aujourd'hui** sont conservées (échéances à venir).

- **Facture sans écheancier :** une seule ligne avec **date_echeance** = date de facture, **montant_echeance** = encours de la facture (si date_facture ≥ aujourd'hui).

7.2 Tri et affichage

- Les lignes sont triées par **date_echeance** croissante.
- L'affichage peut indiquer le numéro d'échéance (ex. « Éch. 1/3 ») pour les factures avec écheancier.

---

8. Exemples de requêtes SQL

8.1 Liste des factures avec encours

```sql
SELECT
    f.*,
    COALESCE(SUM(p.montant), 0) AS total_paye,
    GREATEST(0, f.montant_ht - COALESCE(SUM(p.montant), 0)) AS encours_facture,
    CASE
        WHEN GREATEST(0, f.montant_ht - COALESCE(SUM(p.montant), 0)) <= 0 THEN 'paye'
        ELSE 'en_attente'
    END AS etat
FROM factures f
LEFT JOIN paiements p ON p.facture_id = f.id
WHERE f.client_id = ?
GROUP BY f.id
ORDER BY f.date_facture DESC
```

8.2 Encours par client

```sql
SELECT
    c.id AS client_id,
    c.nom,
    COALESCE(SUM(GREATEST(0, f.montant_ht - COALESCE(sp.total_paye, 0))), 0) AS encours
FROM clients c
LEFT JOIN factures f ON f.client_id = c.id
LEFT JOIN (
    SELECT facture_id, SUM(montant) AS total_paye
    FROM paiements
    GROUP BY facture_id
) sp ON sp.facture_id = f.id
GROUP BY c.id, c.nom
HAVING encours > 0
ORDER BY encours ASC
```

---

9. Évolutions possibles (post-MVP)

- **Rapprochement paiements/échéances** : lier un paiement à une échéance spécifique.

- **Statut par échéance** : à venir / payée / partielle / en retard.
- **Gestion des crédits** : permettre encours négatif (avoir client).
- **Modification/suppression paiements** : avec historique.
- **Échéances non mensuelles** : trimestrielles, annuelles, etc.

Document de référence pour l'implémentation. À mettre à jour si la logique métier évolue.