출자사업 데이터 자동화 - 예외 처리 및 커스터마이징 정리

이 문서는 KVIC 출자사업 PDF 데이터를 자동으로 파싱하여 Google Sheets에 저장하는 과정에서 발생한 복잡한 예외 상황들과 그 해결책을 정리한 것입니다.

---

1. 운용사명 매칭 문제

문제

같은 회사인데 다르게 표기되는 경우가 많음:

- "KB인베스트먼트" vs "케이비인베스트먼트"
- "IBK벤처투자" vs "아이비케이벤처투자"
- "STIC Investments" vs "스틱인베스트먼트"
- "BNK투자증권" vs "비엔케이투자증권"

해결책: 9단계 유사도 알고리즘

단계	검사 항목	유사도 점수
1	정확히 일치	100%
2	정규화 후 일치 (공백/특수문자 제거)	98%
3	부분 문자열 포함	70~95%
4	핵심명 동일 (접미사 제거 후 비교)	95%
5	한글 자모 분해 유사도 ("케이비" → "ㄱㅖㅇㅣㅂㅣ")	90%
6	초성 비교 ("케이비" → "ㅋㅇㅂ")	70%
7	영문 약어 ↔ 한글 발음 양방향 매칭	85%
8	Levenshtein 거리 기반 문자열 유사도	최대 85%
9	공통 접두사 비율	70%

관련 파일

- src/operator-matcher.js

---

2. 접미사만 같은 경우의 오탐 방지

문제

"다성벤처스" vs "효성벤처스" → 유사도 75%로 계산됨

- 완전히 다른 회사인데 "벤처스" 접미사만 같아서 높은 유사도

해결책: 핵심명 유사도 이중 체크

```
// 전체 유사도 60~85%일 때 추가 검증
const core1 = removeCompanySuffix(name1);  // "다성"
const core2 = removeCompanySuffix(name2);  // "효성"
const coreSimilarity = calculate(core1, core2);  // 50%

// 핵심명 유사도 60% 미만이면 → 접미사만 유사한 것으로 판단 → 다른 회사
```

제거 대상 접미사

인베스트먼트, 벤처스, 파트너스, 캐피탈, Investment, Capital, Fund, LLC, Inc 등

---

3. 영문 약어 ↔ 한글 발음 양방향 매칭

문제

초기에는 영문→한글 매칭만 지원 (KB → 케이비)

- "비엔케이투자증권" vs "BNK투자증권" 매칭 실패

해결책: 역매핑 자동 생성

```
// 기존: 영문 → 한글
ABBREV_MAP = { 'kb': '케이비', 'bnk': '비엔케이', 'ibk': '아이비케이' }
```

```
// 추가: 한글 → 영문 (자동 생성)
REVERSE_ABBREV_MAP = { '케이비': 'KB', '비엔케이': 'BNK', '아이비케이': 'I
```

등록된 약어

kb, nh, sk, ibk, bnk, dgb, jb, stic, ds 등

---

4. 법인 표기 차이 처리

문제

"(주)벡터기술투자" vs "벡터기술투자" → 다른 회사로 인식됨

해결책: 정규화 시 법인 표기 제거

```
normalize = name
  .replace(/^\(주\)/, '')
  .replace(/^주식회사/, '')
  .replace(/\(주\)$/, '')
  .trim()
```

---

5. 유사 운용사 웹검색 확인

문제

유사 운용사가 같은 회사인지 다른 회사인지 판단이 어려운 경우

- "아이비케이캐피탈" vs "아이비케이벤처투자" → 같은 IBK 계열이지만 별개 법인

해결책

헷갈리면 WebSearch로 직접 검색하여 확인

---

6. 중복 운용사 병합

문제

이미 중복으로 등록된 운용사를 어떻게 정리할 것인가?

해결책: 병합 워크플로우

```
# 중복 리포트 (유사도 85% 이상 쌍 찾기)
node src/operator-audit.js report

# 병합 실행
node src/operator-audit.js merge <유지ID> <삭제ID> --execute
```

병합 시 자동 처리

1. 삭제할 운용사ID의 신청현황 → 유지할 ID로 변경
2. 약어에 삭제된 운용사명 추가 (검색 편의)
3. 삭제할 운용사 행 삭제

관련 파일

- src/operator-audit.js

---

7. 공동GP 처리

문제

PDF에 "A / B" 또는 "A, B"로 적힌 공동GP를 어떻게 저장할 것인가?

해결책: 개별 행 분리 원칙

```
PDF: "A / B" (1건으로 표기)
       ↓
저장: A (1건, 비고: 공동GP)
      B (1건, 비고: 공동GP)
```

분리 우선순위

1. 줄바꿈 (`\n`)
2. 쉼표 (`,`)
3. 슬래시 (`/`)

통계 형식

```
신청조합 149개, 공동GP 12개(2개조합 10건, 3개조합 2건), 총 신청현황 165건
```

---

8. 파일-출자사업 N:N 관계

문제

파일과 출자사업이 1:1 매칭이 안 됨

케이스 A: 여러 접수파일 + 1개 선정파일

```
접수: FH0081 (일반분야), FH0119 (지역분야)
선정: FH0001 (통합 결과)
→ 지원파일ID: "FH0081, FH0119"
```

케이스 B: 1개 접수파일 + 여러 선정파일

```
접수: FH0082 (문화+영화+해양 통합)
선정: FH0002 (해양만), FH0003 (문화+영화만)
→ 결과파일ID: "FH0002, FH0003"
```

자동 검증

```
// 파일이 다른 출자사업에 이미 연결된 경우 에러 throw
if (linkedFileIds.includes(fileId)) {
  throw new Error(`파일 중복 연결 오류: ${fileId}는 이미 ${pjId}에 연결됨`);
}
```

---

9. 신청현황 중복 방지

문제

같은 운용사가 같은 출자사업에 여러 번 등록됨

해결책: 복합키 중복 체크

중복 체크 기준: `출자사업ID + 운용사ID + 출자분야` 조합

```
const existingKey = `${operatorId}|${category}`;
if (existingApplications.has(existingKey)) {
  // 이미 등록됨 → 스킵
}
```

주의

- 같은 운용사가 **다른 분야**에 신청한 경우 → 별도 신청현황 (정상)
- 같은 PDF를 다른 출자사업으로 재처리할 때 중복 생성 주의

---

10. PDF 이중 파싱 + 비교 검증

문제

PDF 파싱 정확도를 어떻게 보장할 것인가?

해결책: 두 가지 방법으로 파싱 후 비교

```
방법 1: Claude Code 직접 분석 (육안 기반)
방법 2: pdfplumber (표 구조 기반)
          ↓
       결과 비교
          ↓
일치 → 자동 진행
불일치 → Claude Code 결과 우선 (자동)
```

관련 파일

- src/pdf-parser.py
- src/pdf-compare.js

---

11. pdfplumber 파싱 노이즈 처리

문제

pdfplumber가 잘못 파싱하는 경우가 많음

노이즈 유형

1. **제목이 데이터로 파싱됨**: "모태펀드" + "2024년 출자사업" 별도 행으로 인식
2. **회사명 분리 오류**: "MCP Asset Management Co., Ltd." → "MCP Asset Management Co." + "Ltd."
3. **지역명이 데이터 행에 포함**: "유럽/중동 AVP EUR 157.25"에서 분야 감지 후 데이터 스킵

해결책

- Claude Code 분석 결과 우선
- pdfplumber에만 있는 항목 → 무시 (노이즈로 간주)

---

12. 파일명과 PDF 내용 불일치

문제

파일명: "서류심사_결과" (선정결과로 추정) PDF 내용: "접수현황" (실제 내용)

해결책: PDF 내용 우선 원칙

```
- PDF 상단에서 "접수현황", "신청현황" → 파일유형: 접수현황
- PDF 상단에서 "선정결과", "심사결과" → 파일유형: 선정결과
- 파일 시트의 파일유형 컬럼도 함께 수정
```

---

13. 금액 단위 처리

문제

"USD 1,842M"이 그대로 저장됨 → 단위 통일 필요

해결책: 억원/M 단위로 저장

```
원화: 300억원 → 숫자: 300, 통화단위: "억원"
달러: USD 50M → 숫자: 50, 통화단위: "USD(M)"
```

주의

- 문자열 형태가 남아있으면 `parseFloat()` 실패 → 0으로 변경되는 사고 발생
- 반드시 숫자만 저장

---

14. 선정/탈락 판정

문제

접수현황에는 있는데 선정결과에 없으면 탈락인가?

해결책: 약어 확장 + 정규화 기반 매칭

```
// 선정된 운용사 세트 생성 (정규화 + 약어 확장)
const selectedNames = new Set();
for (const s of selected) {
  selectedNames.add(normalizeName(s.name));
  selectedNames.add(normalizeName(expandAlias(s.name)));
}

// 매칭되면 선정, 안되면 탈락
```

---

15. 접수현황에 없는 선정 운용사 처리

문제

선정결과에는 있지만 접수현황 PDF에 누락된 운용사 존재

해결책: 선정결과 순회 후 추가

```
선정결과에서 발견된 운용사가 접수현황에 없으면:
1. 약어 매핑으로 기존 운용사 찾기
2. DB 검색으로 매칭
3. 없으면 새 운용사 생성
4. "선정" 상태로 신청현황 생성
5. 비고: "접수현황 PDF에 미기재, 선정결과에서 확인됨"
```

---

16. 현황 필드 동기화

문제

PDF에 "19개"라고 적혀있는데 실제 저장은 18건 (1건 중복 스킵)

해결책: 현황은 항상 테이블에서 재계산

```
// 파일 현황: 신청현황 테이블에서 계산
await sheets.syncFileStatusWithApplications(fileId);

// 출자사업 현황: "총 171건 (선정 45, 탈락 126)" 형식
await sheets.updateProjectStatus(projectId);
```

---

17. HWP 파일 처리

문제

HWP 파일은 Read 도구로 직접 읽을 수 없음

해결책: Playwright MCP + Google Drive 뷰어

```
1. Google Drive URL로 이동 (browser_navigate)
2. 렌더링 대기 (browser_wait_for)
3. 스크린샷 캡처 (browser_take_screenshot)
4. 이미지 OCR 분석 (Read)
```

여러 페이지 처리

```
- PageDown: 다음 페이지로 이동
- ArrowDown: 페이지 내 스크롤
- 75% 줌 권장 (50%는 OCR 정확도 저하)
```

관련 파일

- src/gdrive-capture.js

---

18. 단계별 처리 원칙 (접수→선정 순서)

문제

두 파일 동시에 보고 바로 선정/탈락 판단 → 프로세스 오류

해결책: 순차 처리 강제

```
1단계: 접수현황 파일 → 모든 신청현황을 "접수" 상태로 저장
2단계: 선정결과 파일 → 선정된 운용사는 "선정", 나머지는 "탈락"으로 업데이트

⚠️ 두 파일을 동시에 보고 바로 선정/탈락을 판단하면 안 됨
```

이유

- 접수현황만 있고 선정결과가 아직 없는 경우 대응 가능
- 데이터 처리 추적 가능

---

19. 검토/승인 워크플로우

문제

파싱 결과를 바로 저장하면 오류 가능성 있음

해결책: 터미널에서 테이블 리뷰

```
[파싱 결과 테이블 표시]
  운용사명 | 출자분야 | 상태 | 비고
  ──────────────────────────────
  A벤처스 | 중진-루키 | 선정 |
  ...

명령어: y(승인), n(취소), e(수정), r(다시보기)
```

관련 파일

- src/review-workflow.js

---

20. API 할당량 관리

문제

Google Sheets API 분당 요청 제한으로 인한 429 오류

해결책

배치 처리

```
// 개별 처리 대신 배치 메서드 사용
const newIds = await sheets.createApplicationsBatch(dataList);   // 
const nameToIdMap = await sheets.createOperatorsBatch(names);    // 
```

자동 재시도

```
- API 할당량 초과 시 1분 대기
- 최대 3회 자동 재시도
- 사용자 개입 없이 자동 진행
```

---

21. 자동화 모드 (사용자 확인 최소화)

문제

너무 많은 상황에서 사용자 확인 요청 → 비효율적

해결책: 자동 진행 모드

상황	처리 방식
이중 파싱 결과 차이	Claude Code 결과 우선 (자동)
pdfplumber에만 있는 항목	무시 (자동)
유사 운용사 85% 미만	신규 등록 (자동)
API 할당량 초과	1분 대기 후 재시도 (자동)

파일 쌍 못 찾음	**사용자 확인 필요**
유사도 85%+ & 핵심명 60%+	**사용자 확인 필요**

---

22. 파일 현황 통계 자동 생성

문제

파일 처리 결과를 한눈에 파악하기 어려움

해결책: 파일 시트 현황 컬럼

```
접수현황: "신청조합 149개, 공동GP 12개(2개조합 10건, 3개조합 2건), 총 신청현황 165건"
선정결과: "총 165개 중 선정 43건"
```

관련 파일

- src/file-summary.js

요약 테이블

#	카테고리	문제	해결책
1	운용사 매칭	영문/한글/약어 불일치	9단계 유사도 알고리즘
2	운용사 매칭	접미사 오탐	핵심명 유사도 이중 체크
3	운용사 매칭	약어 한방향 매칭	양방향 매핑
4	운용사 매칭	법인 표기 차이	정규화 시 제거
5	운용사 매칭	판단 어려움	웹검색 확인
6	운용사 매칭	중복 운용사	병합 워크플로우
7	데이터 처리	공동GP 표기 다양	분리 우선순위
8	데이터 처리	파일-사업 N:N 관계	쉼표 연결 + 자동 검증
9	데이터 처리	신청현황 중복	복합키 중복 체크

10	PDF 파싱	파싱 정확도	이중 파싱 + 비교
11	PDF 파싱	pdfplumber 노이즈	Claude Code 우선
12	PDF 파싱	파일명-내용 불일치	PDF 내용 우선
13	PDF 파싱	금액 단위	억원/M 단위 저장
14	상태 판정	선정/탈락 매칭	약어 확장 + 정규화
15	상태 판정	접수 누락 선정자	선정결과에서 추가
16	상태 판정	현황 불일치	테이블 기반 재계산
17	파일 형식	HWP 파일	Playwright + OCR
18	프로세스	순서 오류	단계별 처리 강제
19	프로세스	검토 필요	터미널 리뷰
20	시스템	API 할당량	배치 + 자동 재시도
21	시스템	확인 요청 과다	자동화 모드
22	시스템	결과 파악 어려움	현황 통계 자동 생성

이 문서는 프로젝트 진행 중 발생한 예외 상황들을 정리한 것으로, 향후 유사 프로젝트 참고용입니다.