IMODEL AI

2026 实用手册

# 从 Excel 到 iModel

六章讲清怎么换、换到哪一步

不写代码已经不稀缺了。

说得清楚这个数是怎么来的，才稀缺。

**iModel 数据科学平台 · 深圳市卓越方达科技有限公司**

网页版与后续更新 www.imodel.org.cn/excel-to-imodel/

IMODEL AI

# 目录

六章，从判断该不该换开始，到把结果发回 Excel 结束。

**关于示例数据。**全书统一用一套虚构的业务数据：订单表（订单号、门店编号、商品编号、数量、下单日期）、门店表（门店编号、门店名称、城市）、商品表（商品编号、商品名称、品类、单价）。你可以照着自己造一份跟着做。

第 一 章

# 2026 年该不该从 Excel 换工具

> **Excel 处理不了大数据，指的通常不是行数超了上限，而是文件还能打开、但已经没法用。**单个工作表的上限是 1,048,576 行，绝大多数人在远远到不了这个数之前就撑不住了 - 十几万行加上几列公式，重算一次要等半分钟，改一个筛选条件又是半分钟。
>
> 但在 2026 年，「算得慢」已经不是最值得换工具的理由。本章讲的是另一个理由，它和 AI 有关。

## 先承认：无代码已经不是理由了

三年前劝人从 Excel 换到数据分析平台，标准说法是「不用写代码也能建模型」。

这个说法现在站不住。Copilot 已经正式进入 Excel，可以用自然语言让它改数据、建图表、写公式；Python 也能通过 Copilot 直接在工作簿里跑，做更复杂的分析和可视化。你想要的大部分计算，现在都可以让 AI 帮你完成。

**所以如果一份迁移指南的第一章还在讲「无需编程」，你可以直接合上。**那是 2022 年的卖点，2026 年它已经是入场券，不是差异点。

## 真正的分界线在哪

有意思的是，这条分界线不需要我来划 - **微软自己在官方文档里已经划好了。**

> 来自微软官方支持文档
>
> 微软建议：任何**需要准确性或可重现性**的任务，应当使用原生 Excel 公式，而不是 AI 生成的结果；涉及**法律、法规或合规性**的场景 - 例如财务报告、法律文档等高风险用途 - 应避免直接采用 AI 生成的输出。
>
> 文档中还有一句更直接的提醒：即使参数完全相同，AI 生成的结果也可能随时间推移而变化。
>
> 出处：Microsoft 支持文档，核实于 2026 年 9 月。

- **AI 生成的结果不保证可重现。**这个月和上个月能不能算出同样的数，厂商自己不打包票。
- **有合规影响的场景不建议用。**而财务报告、监管报送、审计底稿，恰恰是很多人每个月都要做的那类活。

这一段的意义

这不是在说 Copilot 不好用 - 它很好用。这是在说：**好用和可交差是两件事。**探索性地问「这批数据有什么异常」，AI 很合适；每月固定要出、要签字、要留档的那份报表，需要的是别的东西。

## AI 算完之后，三个问题才开始

### 问题一：它对不对

你没法逐格验算。公式看起来合理，数字看起来正常 - 但「看起来正常」和「正确」之间隔着的东西，恰恰是你换工具想解决的。

### 问题二：下个月还能跑出同样的结果吗

这一条微软已经替我们回答了：参数相同，结果也可能变。对探索性分析无所谓，对每月口径必须一致的报表则是硬伤。

### 问题三：审计问起来怎么解释

「这个数是 AI 帮我算的」不是一个可接受的答案。内审、外审、监管要的是**可复现的过程**，不是一个可信的结果。


图 1-1　同一份数据的两条路径。差别不在快慢，在于中间过程是否可见、可复现。

## 工作流回答的正是这三个

iModel　的做法是把处理过程画成一张流程图。读文件、清洗、关联、汇总、导出，每一步是一个节点，节点连成一条线。


图 1-2　工作流的基本形态。第 2 章会带你从零建一条这样的流程。

- **对「它对不对」** - 每一步的中间结果都能点开看。出错时能定位到具体哪一步，而不是面对一个不知道哪里错了的最终数字。
- **对「能不能复现」** - 同样的输入必然得到同样的输出。工作流是确定性的，不存在「这次和上次不一样」。
- **对「怎么解释」** - 流程图本身就是文档。给审计看逻辑直接打开图就行，不需要另写一份说明。

## 那 AI 就没用了吗

有用，但位置变了。

| 年份 | 人做什么 | 工具做什么 |
| --- | --- | --- |
| **2022** | 想清楚逻辑，写公式 | 执行公式 |
| **2026** | 提出问题，判断结果是否合理 | AI 帮忙想怎么做，工作流确定地执行 |

合理的用法是让 AI 帮你想清楚某一步该用什么节点、把一段 Excel 公式翻译成节点组合、解释某个报错是什么意思。想清楚之后，那条工作流是固定的、可检查的、可重跑的 - **AI 参与了构建过程，但没有参与每一次执行。**

**一句话记住**

**AI 负责一次性的创造，工作流负责重复性的确定。**把这两件事交给同一个东西做，就会出现微软文档里警告的那个问题。

## 三个该换的信号

不看行数，看这三个。

信号 01

**每月都在重复同一套操作**

打开三个文件、复制粘贴、删几列、加两个公式、做透视表、截图发出去，下个月再来一遍。特征是**步骤固定但没法记录** - 唯一的记录是你的记忆，人休假了就停摆。

信号 02

**你不敢确认结果对不对**

公式引用了哪些格、中间删过哪一列、某个数是手改的还是算的，时间一长自己都说不清，只能说「我一直是这么做的」。

信号 03

**文件大到不敢动**

超过 50MB 的 xlsx 打开要等、改一格要等、另存为还要等，而且开始偶尔崩溃，你不确定上次保存是什么时候。

图 1-3　三个判断信号。**三个中两个就该换；只中一个，先看下一节。**

## 该用哪个工具

| 场景 | Excel | iModel | 该用哪个 |
|---|---|---|---|
| **临时看一眼几百行数据** | 双击就开 | 要先建工作流 | **Excel** |
| **探索性地问数据有什么规律** | Copilot 很合适 | 需要自己想步骤 | **Excel** |
| **每月重复同一套处理** | 手工重做 | 建一次，之后点执行 | **iModel** |
| **十几万行以上** | 卡顿明显 | 正常处理 | **iModel** |
| **要向审计解释处理逻辑** | 逻辑藏在单元格里 | 流程图即文档 | **iModel** |
| **每月口径必须一致** | 靠人记 | 工作流固定 | **iModel** |
| **做一张给客户看的漂亮表** | 排版能力强 | 需额外配置 | **Excel** |
| **合并几十个格式相同的文件** | 逐个复制粘贴 | 指定文件夹一次读完 | **iModel** |

**第四列是重点。**iModel　不是所有场景都更好，硬把临时看数、探索性分析和排版这类事搬过来，只会更麻烦。

## 什么时候还是该用 Excel

- **一次性的活** - 建工作流的时间比手工做还长。
- **探索阶段** - 还不知道要算什么、想先摆弄数据看看，Excel 加 Copilot 效率更高。
- **需要精细排版的交付物** - 报价单、要打印的报表，Excel 排版能力更强。
- **数据量小且逻辑简单** - 几百行、两三个公式，Excel 又快又直观。
- **协作方全是 Excel 用户** - 你算完还是要发 xlsx 出去。

最后一条值得展开

**换到 iModel 不等于不再用 Excel。**常见的实际形态是：iModel 负责处理和计算，最后一步仍然输出 xlsx 发出去（第 6 章讲怎么做）。两者是上下游关系，不是替代关系。

## 常见的适应难点

### 从单元格思维转到列思维

这是两个工具最根本的结构性差异，也是转换期最容易卡住的地方。

Excel 按单元格算：你在 C2 写 `=A2*B2`，然后往下拖，每一行各算各的。iModel 按列算：你定义一次「新列 = 数量 × 单价」，整列一次算完，没有「往下拖」这个动作，也不需要有。

习惯了 Excel 的人第一次会到处找「往下拖在哪」。找不到不是操作问题，是思路还没转过来。这个转换通常要一两天，属于正常。第 5 章会专门讲。

转过来之后的好处

你的计算逻辑只写了一次。**Excel 里往下拖出来的一万行公式，理论上是一万个独立公式 - 中间任何一行被人手工改过，你不会知道。**这也是前面那个「不敢确认结果对不对」的信号背后的真正原因。

第 二 章

# 第一条工作流：十五分钟跑通

> **Excel 转工作流，指的是把原来在 Excel 里手工点几十下完成的一套操作，改成一条可以重复执行的流程图。**流程图里每个方块是一个节点，负责一件事；节点连起来，数据从左往右流过去，最后一步输出结果。
>
> 这一章不讲原理，只带你跑通一遍。跑通之后再回头看第 1 章的论点，会具体得多。

## 这一章要做的事

**目标：**从一份订单明细里，把 2026 年 3 月的记录挑出来，按门店汇总销售数量，导出成一个新的 Excel 文件。

这是最典型的月度报表活。在 Excel 里你大概会这么做：用筛选选出 3 月的记录，插入数据透视表把门店放行、数量放值，再把结果复制粘贴到新工作簿另存。三步，不难 - 问题是下个月还要再来一遍，而且没人知道你当时筛的到底是哪个日期范围。


图 2-1　本章要建的工作流。节点名是英文，iModel 中文界面里会显示对应的中文名。

## 先认识一个节点

动手之前，花一分钟看清楚一个节点由哪几部分组成。后面所有操作都围着这四样东西转。


图 2-2　一个节点的四个部分。有些节点有多个输入或输出端口，第 3 章的 Joiner 就是。

## 动手之前的两件事

### 一、装好软件

iModel Analytics Studio 是开源版本，免费、不限功能与使用时长。Windows 与麒麟、统信环境都是双击安装。首次启动会让你选一个工作区目录 - 用来存放你所有的工作流，随便选一个空文件夹即可。

### 二、准备一份数据

造两三百行订单明细即可，字段为订单号、门店编号、商品编号、数量、下单日期（覆盖 2026 年 1 月到 4 月）。门店编号控制在五六个以内，这样汇总结果一眼能看完。

> **用自己的真实数据也可以**，但第一次建议用小样本。数据量大、字段多的时候，一旦哪一步配错了，排查会分散你的注意力 - 而这一章的重点是把流程跑通。

## 四个节点走一遍

先新建一个工作流：在左上角的工作流浏览器里右键，选择新建工作流，起个名字，双击打开。你会看到中间一片空白的画布。

找节点的方法

左侧节点库有分类树，但**直接用顶部搜索框输入节点英文名最快**。本章要用到的四个节点，输入 `Excel Reader`、`Row Filter`、`GroupBy`、`Excel Writer` 即可找到。老手基本都用搜索，不翻分类。

01 **读入订单表。**搜索 `Excel Reader` 拖到画布上，双击打开配置，选择订单文件，确认它认出了列名和数据类型，点确定。这时节点下面的灯是黄色：已配置、未执行。右键执行（F7），灯变绿后再右键选最下面那一项，可以看到读进来的数据表。**先看一眼数据对不对，再往下做。**

02 **筛出 3 月的记录。**搜索 `Row Filter` 拖到右边，把两个节点连起来。双击 Row Filter，选择「下单日期」列，设置筛选区间为 3 月 1 日到 3 月 31 日。执行，看行数是不是明显变少。

03 **按门店汇总数量。**搜索 `GroupBy` 接到后面。配置分两步：分组列选「门店编号」，聚合方式选「数量」这一列、函数选求和。执行后得到一张只有几行的表 - 这一步对应的就是 Excel 里的数据透视表。

04 **导出成 Excel 文件。**搜索 `Excel Writer` 接到最后，配置输出路径和文件名，执行。打开那个文件，里面就是你要的汇总表。最后按 Ctrl + S 保存工作流。下个月重跑时，把 Excel Reader 里的文件换成新的，然后执行全部，几秒钟出结果。

## 状态灯：入门阶段最省时间的习惯

每个节点下方有一个小状态灯。看懂它，能解决你入门阶段一大半的困惑。


图 2-3　四种节点状态。排查问题的第一步永远是看灯的颜色。

最常见的一个困惑

「我明明改了配置，结果怎么没变？」**因为改完配置后节点会退回黄灯，需要重新执行。**Excel 里改个公式立刻重算，工作流不是 - 它把「配置」和「执行」分成了两件事。这一点第一天会不习惯，但它正是可复现的来源：什么时候算、算的是哪一版配置，全都是明确的。

## 和原来的三步对照

| 要做的事 | Excel 里 | 工作流里 | 差别在哪 |
| --- | --- | --- | --- |
| **拿到数据** | 双击打开文件 | Excel Reader 节点 | 文件路径记在节点里，换文件只改一处 |
| **筛出 3 月** | 点筛选，勾日期 | Row Filter 节点 | 筛选条件是写下来的，不是点出来的 |
| **按门店汇总** | 插入数据透视表 | GroupBy 节点 | 分组和聚合方式明确可见 |
| **出结果文件** | 复制粘贴，另存为 | Excel Writer 节点 | 输出路径固定，不会存错地方 |
| **下个月再做一遍** | 整套重来 | 换文件，点执行 | 这是全部差别的落点 |

第一次建这条流程，花的时间比在 Excel 里手工做要长 - 这是正常的，也是唯一一次。**从第二个月开始，时间差就反过来了。**

## 常见的适应难点

### 忘记执行上游节点

下游节点是黄灯，双击进去发现配置窗口里选不到列名 - 因为上游还没执行，它不知道会有哪些列。养成「拖一个、连一个、执行一个」的节奏，几乎不会撞上这个问题。

### 改了配置忘了重跑

改完配置节点退回黄灯，结果还是上一次的。看灯的颜色就能发现。

### 找不到「往下拖」

第 5 章会专门讲。简单说：工作流里不需要往下拖，你定义一次列与列的关系，整列一次算完。本章的 GroupBy 已经体现了这个思路 - 你没有对任何一行单独做过操作。

第 三 章

# 表关联：VLOOKUP 怎么办

> **VLOOKUP 替代，指的是把 Excel 里用 VLOOKUP、XLOOKUP 或 INDEX MATCH 完成的表关联，改成用可视化节点来做。**在 iModel 中承担这个职责的是 Joiner 节点，它按列名匹配两张表，并把结果输出为一张新表。
>
> 操作层面的转换不难，半小时就能上手。真正需要花时间的是三处认知上的差异。

> 「我有一张订单表，只有门店编号，没有门店名字。以前我在订单表右边加一列写 VLOOKUP 把名字拉过来。现在这张表有二十多万行，一拖公式电脑就卡半天，而且每次源文件换了我还得重新拖一遍。」

这是 VLOOKUP 最典型的用法：**用一张表里的编号，去另一张表里换取对应的信息**。数据库里管这叫「关联」，Excel 里叫「查找引用」，说的是同一件事。

## 关键事实

| | |
|---|---|
| **Excel 对应函数** | VLOOKUP、XLOOKUP、INDEX + MATCH |
| **iModel 对应节点** | Joiner |
| **匹配依据** | 列名（不是列号，插入列不会影响结果） |
| **输入端口** | 两个（左表、右表） |
| **输出端口** | 可配置为三个：匹配上的、左表未匹配的、右表未匹配的 |
| **一对多时的行为** | 展开为多行（VLOOKUP 只取第一条） |
| **常见搭配节点** | Row Filter、Column Filter、GroupBy |

## Joiner 长什么样

和第 2 章那些单进单出的节点不同，Joiner 有两个输入、最多三个输出。这个形态本身就说明了它和 VLOOKUP 的根本差异。

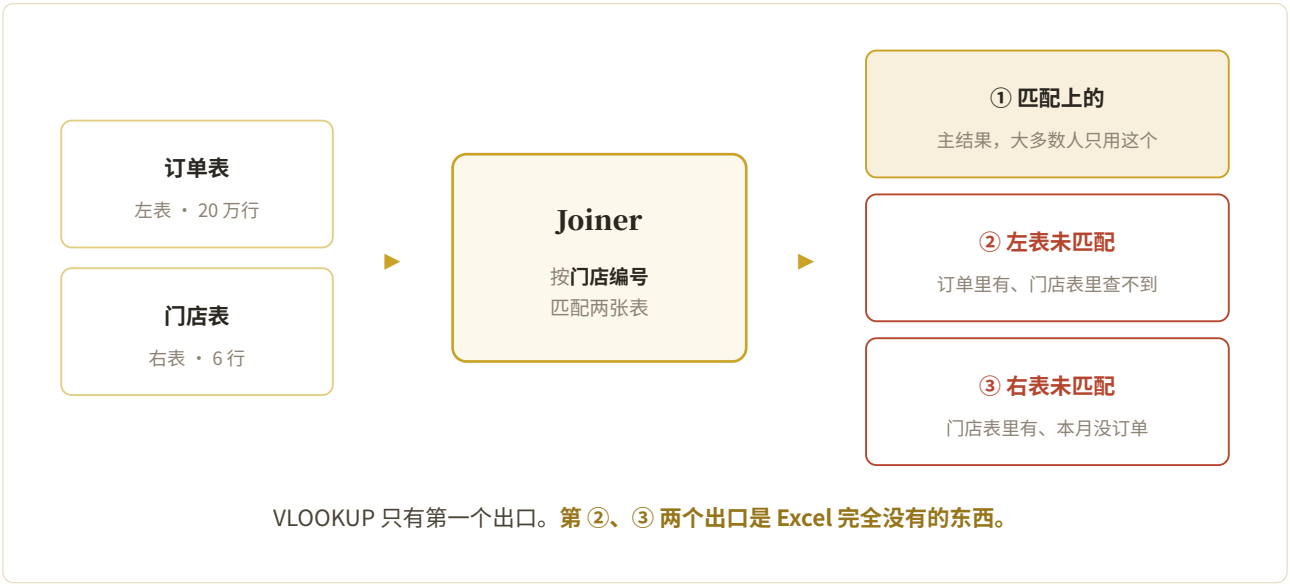


图 3-1 Joiner 的两进三出。第 ② 个出口是这一章最重要的内容。

## 三处需要转过来的认知

Joiner 的操作比 VLOOKUP 简单 - 选两张表、选按哪一列匹配、选连接方式，三步。会卡住的地方不在操作，在下面三个概念差异。

### 错位一：按列名匹配，不按列号

VLOOKUP 的第三个参数是列号，一个数字。`VLOOKUP(A2, 门店表!A:C, 2, 0)` 里的 `2` 意思是「取查找区域的第二列」。这个写法**和列的位置绑死了**。

### 有人在门店表里插了一列

VLOOKUP · 按位置取

| A | B ←取这列 | C |
|---|---|---|
| 门店编号 | **城市** | 门店名称 |
| S001 | **深圳** | 南山店 |

**公式还在跑，返回的却是「深圳」而不是「南山店」 - 而且不报错。**

### 同样插了一列

Joiner · 按列名取

| 门店编号 | 城市 | 门店名称 |
|---|---|---|
| S001 | 深圳 | **南山店** |
| S002 | 广州 | **天河店** |

**你指定的是「门店名称」这个名字，不是「第二列」这个位置。插多少列都不影响。**

图 3-2　按列号取 vs 按列名取。前者是 VLOOKUP 最常见的静默出错来源。

## 错位二：一对多会变成多行

这是转换期最容易被当成 bug 的地方。VLOOKUP 不管右表里有几条匹配，永远只返回**找到的第一条**，左表一行还是一行。Joiner 不是 - 右表里如果有三条记录匹配上，输出就是三行。

### 右表有重复

S001 出现了两次

| 编号 | 名称 |
|---|---|
| **S001** | 南山店 |
| **S001** | 南山店(旧) |
| S002 | 天河店 |

### VLOOKUP

只取第一条，不告知

| 订单 | 名称 |
|---|---|
| SO-001 | 南山店 |

**一行进，一行出。你不知道还有另一条记录存在。**

### Joiner

展开为多行

| 订单 | 名称 |
|---|---|
| SO-001 | **南山店** |
| SO-001 | **南山店(旧)** |

**一行进，两行出。你立刻发现源数据有重复。**

图 3-3　一对多的两种处理。行数变多不是出错，是问题被显示出来了。

为什么这是对的

**行数变多不是 Joiner 出错，是它把 VLOOKUP 一直在替你隐藏的问题显示出来了。**真正该做的是回头处理源数据的重复。

## 错位三：没匹配上的记录不会消失

这一条对做审计和对账的人最重要。

VLOOKUP 找不到时返回 `#N/A`。因为一整列 `#N/A` 很难看，标准做法是套一层 `IFERROR` 变成 0 或空白。**然后这些没匹配上的记录就静默消失了。**月底数对不上，你知道少了，但不知道少在哪 - 因为它们已经被处理成 0，混在正常数据里。

Joiner 把没匹配上的记录单独输出到第 ② 个端口（见图 3-1），**你能直接点开看是哪些。**

> **这一条为什么值钱**
>
> 「这个数是怎么来的」是审计、复核、交接时唯一有用的东西。一个把异常静默吞掉的流程，出问题时你只能重头查；一个把异常单独放出来的流程，出问题时你直接看那一路的输出就行。

## 在 iModel 里的五步

01 拖入两个 Excel Reader，分别读订单表和门店表，各自执行一次确认数据读进来了。

02 拖入 Joiner，订单表接上面的输入端口（左表），门店表接下面的（右表）。

03 双击 Joiner，在匹配条件里选择：左表的「门店编号」对应右表的「门店编号」。

04 选连接方式。想保留全部订单就选左连接；只要能匹配上的就选内连接。

05 在输出设置里勾上把未匹配的行输出到单独端口，然后执行。**执行完先看第 ② 个端口有没有数据** - 有数据就说明源数据有问题，先处理它。

第五步是这一章最想让你养成的习惯。**先看未匹配端口，再看主结果** - 这个顺序和 Excel 里「先看结果对不对」是反的，但它能省掉大量后期排查。

## 该用哪个

| 场景 | Excel 做法 | iModel 做法 | 该用哪个 |
| --- | --- | --- | --- |
| **临时查一个编号对应什么** | 直接 VLOOKUP | 要建工作流 | **Excel** |
| **每月固定关联同几张表** | 每次重拖公式 | 建一次，之后点执行 | **iModel** |
| **二十万行以上** | 拖公式明显卡顿 | 正常处理 | **iModel** |
| **需要知道哪些没匹配上** | 套 IFERROR 后就看不见了 | 单独端口输出 | **iModel** |
| **右表可能有重复记录** | 只取第一条，不告知 | 展开为多行，问题暴露 | **iModel** |
| **源表列的位置常变动** | 列号写死，容易错 | 按列名匹配，不受影响 | **iModel** |
| **一次性核对，用完就扔** | 几分钟搞定 | 建流程反而慢 | **Excel** |

## 常见的适应难点

### 「行数怎么变多了」

见图　　3-3。多出来的那几行，对应的是右表里有重复记录的编号。Excel　　时代这个问题一直存在，只是 VLOOKUP 帮你把它藏起来了。

**判断方法：**执行后对比输入输出的行数。数字不一致就去查右表的匹配列有没有重复值。这个习惯养成之后，你会开始发现一些以前不知道存在的数据质量问题 - 这算好事，虽然刚开始会有点烦。

第 四 章

# 筛选、去重与分组汇总

**数据透视表替代，指的是把 Excel 里用透视表完成的分组与汇总，改成用节点来做。**Excel 的透视表把「行」「列」「值」三件事打包在一个界面里；工作流把它拆开了 - 只要行和值用 GroupBy，需要把某个字段铺成列才用 Pivoting。

本章还会讲筛选和去重。这两件事在 Excel 里都是「改变当前视图」，在工作流里都是「产生一张新表」，差异比看起来大。

## GroupBy 还是 Pivoting

这是从 Excel 转过来最容易搞混的一处。先看结果形态就明白了。

### 原始数据

一行一条明细

| 门店 | 品类 | 数量 |
|---|---|---|
| S001 | 饮料 | 30 |
| S001 | 零食 | 20 |
| S002 | 饮料 | 45 |
| S002 | 零食 | 15 |

四行明细，要按门店看合计。

### GroupBy 的结果

长表 · 分组一列 + 值一列

| 门店 | 数量合计 |
|---|---|
| S001 | 50 |
| S002 | 60 |

列数不变，行数变少。做自动化流程时更安全。

### Pivoting 的结果

宽表 · 把品类铺成列

| 门店 | 饮料 | 零食 |
|---|---|---|
| S001 | 30 | 20 |
| S002 | 45 | 15 |

这才是 Excel 透视表的形态。列数随数据变。

图 4-1　同一份数据的两种汇总结果。区别只在于「有没有把某个字段铺成列」。

怎么选

**问自己一句：结果表的列，是不是由数据内容决定的？**

如果列固定（门店、数量合计），用 **GroupBy**。如果列要随数据变（有几个品类就出几列），用 **Pivoting**。

实际项目里 GroupBy 用得远比 Pivoting 多。因为 Pivoting 的列数不稳定 - 下个月多了一个新品类，结果表就多一列，下游节点可能因此报错。**做自动化流程时，长表比宽表安全。**

做法上，GroupBy 的配置分两步：先选**分组列**（按什么分组），再选**聚合列和聚合方式**（对哪一列做什么运算）。Pivoting 多一步，还要选**透视列**（哪个字段铺成列）。

**一个 Excel 用户容易漏的能力：**GroupBy 可以对同一列同时做多种聚合，也可以对不同列做不同聚合 - 比如门店分组后，数量求和、订单号计数、单价取平均，三件事在一个节点里配完。Excel 透视表里要拖三次值字段才行。

## 筛选：和 Excel 的根本差异

Excel 里点筛选，是把不符合条件的行**藏起来**。原数据一行没少，取消筛选就回来了。工作流里的 Row Filter 不是藏，是**输出一张只包含符合条件那些行的新表**。


图 4-2　两种筛选模型。工作流里「数据没了」是错觉，它在上游节点手里。

这个差异带来一个实际好处：**筛选条件是写下来的，不是点出来的。**Excel 里你上个月勾了哪几个日期，三个月后没人记得；工作流里那个条件明明白白写在配置里。

只按一列筛用 **Row Filter**；要写「A 列等于甲**或** B 列大于乙」这类组合条件，用 **Rule-based Row Filter**。

## 去重：被删掉的行去哪了

Excel 的「删除重复项」是破坏性的：点下去数据就没了，只能靠撤销找回来，而且它不会告诉你删掉的是哪些。

**Excel · 直接删**

删完就没了

| 编号 | 名称 |
| --- | --- |
| S001 | 南山店 |
| S002 | 天河店 |

**被删的那条长什么样？两条到底哪里不一样？无从得知。**

**工作流 · 保留并标记**

看清楚再决定

| 编号 | 名称 | 标记 |
| --- | --- | --- |
| S001 | 南山店 | 保留 |
| S001 | 南山店(旧) | 重复 |
| S002 | 天河店 | 唯一 |

**三种状态一目了然，可以先判断再处理。**

图 4-3　去重的两种模式。第一次处理某份数据时，强烈建议先用右边这种跑一遍。

**建议的用法**

先用标记模式跑一遍，看清楚重复到底是什么样的 - 是完全一样的两行，还是只有关键字段重复而其他字段不同？这两种情况的处理方式完全不同。**直接删掉的问题在于：你不知道自己删了什么。**

这一点和第 3 章讲的 Joiner 未匹配端口是同一个思路：**让异常可见，而不是让它静默消失。**

## 该用哪个

| 场景 | Excel 做法 | iModel 做法 | 该用哪个 |
| --- | --- | --- | --- |
| **临时看某个门店的合计** | 筛一下就看到了 | 要建两个节点 | **Excel** |
| **每月固定出汇总表** | 每次重做透视表 | 建一次，之后点执行 | **iModel** |
| **结果要交给下一步继续算** | 透视表不好当数据源 | 输出就是标准数据表 | **iModel** |
| **需要记录筛了什么条件** | 点选记录不下来 | 条件写在节点配置里 | **iModel** |
| **想知道去重删掉了哪些** | 删完就没了 | 可保留并标记 | **iModel** |
| **拖着字段来回试不同角度** | 透视表拖拽很快 | 改配置要重跑 | **Excel** |
| **一次做多种聚合** | 要拖多次值字段 | 一个节点配完 | **iModel** |

探索阶段拖着字段来回试，Excel 透视表的交互效率仍然更高；**固定下来、要重复出的报表，才值得搬到工作流里。**

## 常见的适应难点

### 「数据被我删掉了吗」

**没有。**点开上游那个节点的输出，完整数据还在。工作流里每个节点都保留自己的输出结果，删除是不可能发生的事 - 除非你去改源文件。这也是工作流敢让你随便试的原因。

### Pivoting 出来的列名很奇怪

输出列名通常是「透视值 + 聚合方式」拼起来的，这是默认命名规则，不是配错了。结果要发给别人看，在后面接一个 Column Renamer 改成正常的名字。

### 分组之后其他列不见了

GroupBy 只输出分组列和聚合列。这是设计如此 - 一个门店有二十条订单，那二十条的订单号该显示哪一个？确实需要保留某列，就把它也加进聚合，选「拼接」或「取第一个」这类方式。

第 五 章

# 公式：从单元格思维到列思维

> **Excel 公式转换，指的是把写在单元格里、靠往下拖复制的公式，改成写在节点里、作用于整列的表达式。**Excel 里所有计算都叫「写公式」；工作流把它拆成三类，交给三个不同的节点：算数字、处理文本、做条件判断。
>
> 这是本手册概念差异最大的一章。前面几章都提到过它，这里把它讲透。

> 「我在订单表右边加一列写 `=D2*E2` 算金额，然后双击往下填满。到了 iModel 里，我加了个节点写好公式，然后就懵了 - 往下拖在哪？我只算了第一行吧？」

没有往下拖，也不需要。**你写那一次，整列就已经算完了。**这句话听起来简单，但真正接受它需要一两天 - 因为它推翻的不是一个操作，是一整套工作方式。

## 往下拖去哪了


图 5-1　同一个计算的两种形态。差别不在打字多少，在于逻辑存在几份。

这就是第 1 章那个论点的技术根源

「你不敢确认结果对不对」这个信号，追到底就是这里。**Excel 里往下拖出来的一万行公式，理论上是一万个各自独立的公式。**中间任何一行被人手工改过、或者拖的时候漏了几行，从表面看不出来。工作流里逻辑只写一次，也就只有一处可以出错、一处需要检查。

表达式里引用列，写法是 `$列名$`。不需要手打 - 配置窗口左侧有列名清单，双击就插入到表达式里，不会拼错。

## SUM 其实是两件事

这是 Excel 用户从没意识到、但转过来后必须分清的一件事。Excel 的 `SUM` 有两种用法，因为都叫 SUM，所以没人觉得它们不一样。

| 门店 | 数量 | 赠品数 |
|---|---|---|
| S001 | 30 | 2 |
| S002 | 45 | 0 |
| S003 | 28 | 5 |

**→ 横向**

**同一行内几列相加**

数量 + 赠品数 → 用 **Math Formula**

**↓ 纵向**

**一整列跨行求和**

数量这一列求和 → 用 **GroupBy**

同样叫 SUM，方向不同，节点不同。**这是转换期最容易混的一处。**

图 5-2　SUM 的两个方向。Excel 用了十年也没人分过，但工作流必须分。

一句话记住

**结果还是每行一个值 → 用 Math Formula。结果行数变少了 → 用 GroupBy。**

## 三个节点的分工

Excel 里不管算什么都是写公式。工作流按**数据类型**分了三个节点，这是转过来第一件要记住的事。


图 5-3　三个节点的分工。判断方法：你要的结果是什么类型，就用哪一个。

**为什么要分开？**分开之后每个节点只做一类事，配置界面就能针对性地给出可用函数清单，找起来快得多。代价是你得先判断「我这个计算属于哪一类」 - 一两天就习惯了。

## Rule Engine 值得单说

Excel 里做多层判断要嵌套 IF：`=IF(A2>1000,"大单",IF(A2>500,"中单","小单"))`，层数一多就没法读。Rule Engine 是一行一条规则，从上往下匹配，命中就停：

| 写法 | 含义 |
| --- | --- |
| `$金额$ > 1000 => "大单"` | 金额大于 1000 的，标为大单 |
| `$金额$ > 500 => "中单"` | 剩下的里面，大于 500 的标为中单 |
| `TRUE => "小单"` | 其余全部标为小单（兜底规则） |

**三条规则平铺，比三层嵌套 IF 好读得多**，加一档只需要多写一行。这是从 Excel 转过来后少数会觉得「新工具确实更好用」的地方。

## 常用计算的对照

| 要做的事 | Excel 写法 | 用哪个节点 | 表达式 |
|---|---|---|---|
| **两列相乘** | `=D2*E2` | Math Formula | `$数量$ * $单价$` |
| **同行几列相加** | `=SUM(D2:F2)` | Math Formula | `$A$ + $B$ + $C$` |
| **整列求和** | `=SUM(D:D)` | **GroupBy** | 聚合方式选求和 |
| **四舍五入** | `=ROUND(D2,2)` | Math Formula | `round($金额$, 2)` |
| **文本拼接** | `=A2&B2` | String Manipulation | `join($省$, $市$)` |
| **替换字符** | `=SUBSTITUTE(…)` | String Manipulation | `replace($编号$,"-","")` |
| **截取前几位** | `=LEFT(A2,3)` | String Manipulation | `substr($编号$, 0, 3)` |
| **条件判断** | `=IF(…)` | Rule Engine | `$金额$>1000 => "大单"` |
| **跨表查值** | `=VLOOKUP(…)` | **Joiner** | 见第 3 章 |

**注意最后两行。**整列求和和跨表查值在 Excel 里也是「写公式」，但在工作流里它们不属于公式类节点 - 前者是聚合，后者是关联。这是分类方式的差异，不是功能缺失。

## 该用哪个

| 场景 | Excel | iModel | 该用哪个 |
| --- | --- | --- | --- |
| **临时算一下某个值** | 找个空格就算 | 要建节点 | **Excel** |
| **整列同一套计算** | 写一次往下拖 | 写一次作用整列 | **iModel** |
| **多层条件判断** | 嵌套 IF，难读 | 规则平铺，一行一条 | **iModel** |
| **需要引用上一行的值** | 写 `=A2-A1` 即可 | 需要专门的节点 | **Excel** |
| **计算逻辑要交接给别人** | 藏在单元格里 | 写在节点配置里 | **iModel** |
| **要保证每行算法一致** | 无法保证 | 结构上不可能不一致 | **iModel** |

**第四行值得注意。**引用相邻行是 Excel 的强项 - 环比、累计、和上一行做差，写起来非常自然。工作流里这类计算需要专门的节点先把上一行的值取到本行来，绕一道弯。**这是 Excel 少数在结构上更方便的场景之一。**

## 常见的适应难点

### 找不到「往下拖」

写完表达式点确定、执行，结果就已经是整列了。**建议第一次用五行数据跑一遍** - 五行能一眼看完，比在几万行里找证据快得多。

### 不知道该用哪个节点

看你要算的结果是什么类型：算出来是数字用 Math Formula，是文字用 String Manipulation，是分类标签用 Rule Engine。

### 想引用上一行的值

Math Formula 做不到 - 它逐行独立计算，每一行看不到别的行。这不是缺陷，是设计取舍：正因为每行独立，整列计算才能保证一致。

### 类型不对导致报错

Math Formula 只处理数值列。某列从 Excel 读进来时被识别成文本（常见于带空格或带单位的数字），用它就会报错。在前面接一个 String to Number 转换类型，或回到 Excel Reader 的配置里改列类型。这是从 Excel 读数据时最高频的一类问题。

第 六 章

# 输出：把结果发回 Excel

> **自动生成 Excel 报表，指的是让工作流在跑完之后直接产出一个 xlsx 文件，不需要人工复制粘贴。**iModel 提供三种方式：新建一个文件、往已有文件里追加工作表、或者只更新已有文件中特定单元格的内容。
>
> 第三种是本章的重点 - 它能让你保住 Excel 模板里辛苦做好的排版和公式。

> 「流程都跑通了，数字也对，可导出来的 xlsx 是光秃秃一张表 - 没有公司抬头，没有配色，没有边框。我原来那个模板做得挺好看的，每次改数字就行。现在难道要我每个月手工往模板里贴一次？」

不用。这正是第三种输出方式要解决的问题。

## 关键事实

| | |
|---|---|
| **新建 Excel 文件** | Excel Writer |
| **追加工作表** | Excel Writer，写入选项选追加 |
| **更新已有单元格** | Excel Cell Updater |
| **导出 CSV** | CSV Writer |
| **默认输出样式** | 无格式的朴素表格：无配色、无边框、无列宽调整 |
| **能否保留原模板格式** | 能，用 Excel Cell Updater 只更新数据区 |
| **一次写多个工作表** | 可以，Excel Writer 支持多个输入端口，每个对应一个工作表 |

## 三种输出方式

方式一

### 新建文件

每次生成一个全新的 xlsx。最简单，也最常用。

**适合：**结果表是独立交付物，不需要嵌进现有报表。

方式二

### 追加工作表

往已存在的工作簿里加一个新 sheet，原有内容不动。

**适合：**一个文件里按月份或按分公司分 sheet 累积。

方式三

### 更新单元格

只改写指定位置的单元格，其余一切原样保留。

**适合：**有做好的模板，每月只换数据。

图 6-1　三种输出方式。前两种用 Excel Writer，第三种用 Excel Cell Updater。

## 模板法：保住你辛苦做好的排版

先说清一件事：**Excel Writer 输出的是朴素表格** - 没有配色、没有边框、没有调好的列宽。这不是配置问题，是它的定位就是写数据，不是做排版。

所以想要好看的报表，正确做法不是让工作流去画格式，而是**让 Excel 继续负责它擅长的那部分**。

**模板文件（做一次）**

| 公司抬头 · 月度销售汇总 | | |
|---|---|---|
| 门店 | 数量 | 金额 |
| | | |
| | | |
| 合计（公式） | | |

→ 只更新数据区

**每月生成的成品**

| 公司抬头 · 月度销售汇总 | | |
|---|---|---|
| 门店 | 数量 | 金额 |
| S001 | 50 | 4200 |
| S002 | 60 | 5100 |
| 合计（公式） | 110 | 9300 |

深色格是模板里**固定不动**的部分，白格是工作流每月**写进去的数据**。
模板里的合计公式会按新数据自动重算。

图 6-2　模板法。格式做一次，之后每月只换数据区的内容。

### 具体怎么做

01 在 Excel 里做好模板：抬头、表头、配色、边框、列宽，以及合计行的公式。数据区留空。

02 把模板另存一份作为「母版」，每月流程从母版复制一份出来再更新，这样母版永远干净。

03 工作流末尾用 Excel Cell Updater，指定要写入的单元格地址和对应的值。

04 执行。打开生成的文件，格式全在，数据是新的，合计公式自动重算。

为什么这个方案好

**它把两件事分给了各自擅长的工具。**数据处理的准确性和可复现性交给工作流，视觉呈现交给 Excel。谁都没被强迫做自己不擅长的事。而且模板是可以交接的：接手的人打开模板就知道这张报表长什么样、哪些格子是自动填的。

**关于更复杂的格式化：**如果需要工作流直接生成带完整样式的 Excel（条件格式、动态配色等），社区有相关扩展可以做到。这类扩展不属于核心节点，是否可用请在你自己的环境中确认。**但在大多数场景下，模板法更省事也更稳** - 格式改起来在 Excel 里点几下就行，不用回去改工作流。

## 该用哪个

| 你的情况 | Excel 手工做 | 工作流输出 | 该用哪个 |
|---|---|---|---|
| **一次性出个数发给同事** | 复制粘贴就行 | 要配输出节点 | **Excel** |
| **每月固定出同一张报表** | 每次重新贴 | 点执行，文件就在那 | **iModel** |
| **有做好的模板要复用** | 手工往模板里贴 | Cell Updater 自动填 | **iModel** |
| **要拆成几十个分店文件** | 拆到手软 | 循环批量生成 | **iModel** |
| **报表格式经常要调整** | 直接在 Excel 里改 | 改模板即可，流程不动 | **两者配合** |
| **要做复杂的图表和排版** | Excel 的强项 | 需要额外扩展 | **Excel** |
| **结果要给系统调用** | 做不到 | 可导出 CSV 或对接接口 | **iModel** |

**注意第五行写的是「两者配合」。**这是全书最想让你记住的一句话 - 到了输出这一步，最好的答案往往不是二选一。

## 常见的适应难点

### 导出的文件没有格式

Excel Writer 写的是数据不是样式，想要格式用模板法。第一次遇到会以为是配置漏了什么，其实不是。

### 文件被占用，写不进去

目标 xlsx 正在 Excel 里打开着，工作流就写不进去。**执行输出节点前先关掉那个文件** - 这是最高频的一个低级错误，几乎每个人都会撞一次。

### 数字变成文本，或日期变成数字

写入 Excel 时列的类型可能和预期不一样，最常见的是日期显示成一串数字。输出前用类型转换节点确认好类型，或者在模板里预先把那一列的单元格格式设成日期。

### 每月文件名要带日期

固定文件名会被覆盖，历史留不下来。可以在流程里生成带日期的文件名传给输出节点，属于稍进阶的用法，但一旦配好就再不用管。

结　语

# 六章走完之后

第 1 章开头说过一句话：**换到 iModel 不等于不再用 Excel。**走到这里，这句话应该具体了。


图 7-1　全书的落点。这个分工比「全面替代」诚实，也更容易在团队里推行。

**回到第 1 章的论点**

AI 让算出一个数变得很容易，但说清楚这个数怎么来的反而更难了。**工作流的价值不在于算得快，在于算完之后你能解释。**而 Excel 依然是那个把结果送到人手上的工具 - 这一点在可预见的将来都不会变。

## 接下来做什么

如果你是跟着这六章读下来的，现在建议做一件具体的事：

01 **挑一个你每月都要做、步骤最固定、最烦的活。**不要挑最复杂的，挑最烦的。

02 **照着第 2 到第 6 章搭一遍。**会卡住，卡住的地方就是你真正需要学的。

03 **和原来 Excel 的结果逐项比对**，确认口径完全一致。这一步别省。

04 **连续跑两个月。**第一个月省不了时间，第二个月才看得出差别。

两个月之后再决定要不要搬第二个。这个节奏比一口气全搬要慢，但成功率高得多。

## 继续深入

本手册讲的是「为什么和什么时候」。每个节点的参数怎么配、每条报错什么意思，在网站上有完整的中文资料：

**节点手册** www.imodel.org.cn/knime/nodes/
**报错排查** www.imodel.org.cn/knime/errors/
**入门教程** www.imodel.org.cn/tutorials/
**本手册网页版** www.imodel.org.cn/excel-to-imodel/

---

**关于 iModel。**iModel 数据科学平台由深圳市卓越方达科技有限公司研发，基于 KNIME 开源内核二次开发，中文化、信创适配与企业级增强部分为自主研发，源代码可向客户提供审查。iModel Analytics Studio 为开源版本，免费、不限功能与使用时长。

**关于运行环境。**麒麟 V10、V11 与统信 UOS 的 x86-64 版本已完成测试并在客户现场实际部署，处理器方面支持海光 C86 等 x86-64 平台。ARM 架构（鲲鹏、飞腾）的适配在规划中，具体环境建议在选型阶段确认。