TABLE OF CONTENTS

Contents

01	How Suno Works & The Prompt Formula	FOUNDATIONS
02	61 Niche Genre Deep-Dives	REFERENCE
03	The Curated Style Tag Reference	REFERENCE
04	Suno v6 Features — The Advanced Playbook	FEATURES
05	Lyric Craft for Suno	TECHNIQUE
06	Troubleshooting — 18 Failure Modes	FIX WORKFLOWS
07	Album Mode — Narrative Arcs & Sonic Cohesion	COMPOSITION
08	The Monetization Chapter	BUSINESS

A	The Prompt Pack	REFERENCE
B	The Glossary	REFERENCE
C	The Cheat Sheet	REFERENCE
D	The Index	REFERENCE
E	Changelog & Lifetime Updates	REFERENCE

COLOPHON

WHAT'S NEW · NOVEMBER 2026 EDITION

Keep your prompt yours.

Suno changed several load-bearing things after v6 launched, and two of them change what you should do on your very next song. Every fact below was re-read on Suno's own pages on 26 October 2026; the full dated changelog is in Appendix E.

0

Set Variety to 0

v6's Variety slider works "by adjusting and updating your style prompts": above 0 it rewrites the Style box you wrote. Suno's own advice is 0 for full control. *(Chapter 4, section 9.)*

↓

Rights follow the download

"Songs downloaded while subscribed are granted commercial use rights." Download what you might release before you cancel. Stems count as part of a song's one download. *(Chapter 8.)*

8′

Eight minutes, Max Mode

Every v6 model generates up to 8 minutes in one go, so the genre recipes stop telling you to splice. Max Mode spends more credits to get the final take right. *(Chapters 2 and 4.)*

✎

Edit by describing

Change part of a song by saying what to change, or fix a single lyric without rebuilding it, and when Replace Section is still the better move. *(Chapter 4, section 4.)*

3

The consistency stack, again

Personas now live inside the Voices menu as Style Personas; custom models run on v6. What sits on Pro and what is Premier-only, from Suno's pricing page. *(Chapter 4.)*

+4

New since September

Mixed inputs in Simple mode, Speech (beta) for spoken word over music, Suno Sounds for one-shots and loops, and adding vocals or instrumentals to an existing song. *(Chapter 4.)*

Your purchase includes lifetime updates. When Suno ships a new model or a major feature, you'll get the next edition by email — same download link, fresh file.

- **Chapter 8 — Monetization.** Distribution, sync licensing, copyright, taxes. The business side nobody else is covering for Suno creators specifically.
- **Appendix A — The Prompt Pack.** Every copy-paste prompt from the book in one place.

This is the **November 2026 edition**. The September edition was a real rewrite — not a date change. Suno replaced its whole model line with **v6 on 9 September 2026** — v5, v4.5 and v4 are no longer selectable — and the tools around it moved a lot since the last printing, and the spine of everything that follows is a single shift: **v6 rewards description, not parameters — and now reads it per section.** The page just before this one summarizes what's new; the full dated changelog is in Appendix E. Throughout, anything we ***can't*** confirm against Suno's own documentation is flagged as ***community-tested*** — so you always know what's verified and what's reported.

Make it yours

This book teaches you to describe sound. It does not teach you to copy anyone, and we want to say that plainly before you start.

- **Don't imitate a named artist.** Suno doesn't allow prompts for specific artists or songs, and in its own words it exists "to help people create new, original music, not to trade on anyone's name or likeness." We agree with that line and we hold this book to it. Earlier editions had a chapter that turned forty-one artists into descriptor recipes. We took it out in this edition, because a recipe for sounding like one person is still a way around that rule.
- **Never use someone else's lyrics.** Not a line, not a chorus as a tribute. Write your own words, or use words you have the rights to.

- **Only use a voice you have the right to use.** Your own, or one whose owner said yes.
- **Learn from everyone, then make something new.** Genres belong to everyone. The genre chapter names artists so you know what to listen to, never what to type. Every musician learns by listening to the music they love. Handing in a copy of it is a different thing.
- **Be honest about how you made it.** When you release AI-assisted music, follow your distributor's disclosure rules and never present it as someone else's work.

Everything in the following chapters is built on those five lines.

One promise

We update this book. When Suno ships a new model or changes something load-bearing, we rewrite the affected chapters and send every buyer the new edition for free. The book you hold today is the **November 2026 edition**. If you're reading it a year from now and something looks stale, email us — you're owed the latest. That's not a marketing line; it's why you'll get this exact file refreshed in your inbox.

What changed in the November 2026 edition

- **The artist chapter is gone.** It translated forty-one named artists into descriptor-only prompts. However careful the descriptors, its job was to help you sound like someone else, and that is not what this book is for. The chapters after it moved up one number.

CHAPTER 01 · FOUNDATIONS

How Suno Works & The Prompt Formula

Every Suno generation starts the same way: you hand the model a string of descriptors and it tries to build something sonically coherent out of them. The difference between a generic track and one that actually sounds *specific* lives in how you structure that string. This chapter is the single most-read page in the entire book — the part you come back to every time you're about to hit generate.

The model in plain language

Suno v6 is trained on enormous amounts of music paired with text descriptions. When you feed it a prompt, it's searching its latent space for the region that matches the descriptors you gave. The more specific the descriptors, the tighter the search; the more vague, the more it averages toward a middle-of-the-road interpretation. "Sad song" pulls from a space that contains every sad song ever made — you'll get something generic because the average of every sad song is generic. "Acoustic folk, fingerpicked guitar, tender female vocals, intimate room mic, 85 BPM" pulls from a narrow corner of that space. That's why specificity is the entire game.

The prompt formula

THE STRUCTURE

```
[genre] + [mood] + [instrumentation] + [vocal style] +
[production texture] + [tempo / era]
```

Eight to fourteen tags total. Fewer than eight and Suno improvises to fill the gaps. More than fourteen and it starts averaging them into mush. Lead with the genre — that's the anchor everything else attaches to. The other five categories can appear in any order, but genre always goes first.

The specificity ladder

Here's the same creative idea at four levels of specificity. Watch what happens to the output:

> **Level 1 — Vague:** *sad song*. Suno generates nothing distinctive. The average of every sad song.

> **Level 2 — Genre:** *indie folk, melancholy*. Better. A narrower space, but still thousands of possibilities.

> **Level 3 — Instrumentation + vocal:** *indie folk, melancholy, fingerpicked acoustic, tender female vocals*. Now Suno has a shape to work with.

> **Level 4 — Full formula:** *indie folk, melancholy, fingerpicked acoustic, upright bass, tender female vocals, intimate mic, wooden warmth, 85 bpm*. Suno knows exactly where to land.

Why most prompts fail

Three failure modes account for 80% of bad Suno output. The rest of this book exists to prevent them:

1. **Category gaps.** You specified a genre and mood but skipped vocal style and production. Suno had to guess two out of five categories, so it averaged them.
2. **Conflicting tags.** You stacked "heavy metal" and "smooth jazz" hoping for fusion. Suno tried to satisfy both, failed, and defaulted to the more represented style.
3. **Marketing-label tags.** You added words like "professional", "catchy", "hit song", or "Billboard top 40". These have no sonic meaning to the model. They're noise in the prompt.

The single most valuable habit in Suno prompting is removing any tag that doesn't describe an actual

sound.

Everything that follows — 61 niche genre deep-dives, the curated tag reference, v6 feature walkthroughs, lyric craft, troubleshooting — builds on this one principle. Specificity is the prompt. Everything else is presentation.

CHAPTER 02 · REFERENCE

61 Niche Genre Deep-Dives

Sixty-one niche genres, dissected. Each spread gives you the sonic blueprint, the history, and three tested Suno prompts — core, darker, brighter. Use the prompts as your jumping-off point, not your endpoint. The goal isn't to clone a genre. It's to understand it well enough to break it on purpose.

> ***How to read the "The move" call-outs in this chapter.*** *They are creator-tested heuristics drawn from real generation sessions — ours and the community's — not documented Suno behaviour. Suno does not publish what its model does and does not respond to, so nobody can. Treat each one as a strong starting guess that has worked repeatedly, not as a guarantee, and trust your own ears over any sentence in this book if the two disagree.*

Phonk (drift)

TL;DR — Slowed, foggy, Memphis-rooted trap designed to soundtrack tire smoke and tunnel lights.

BPM — 138–150 BPM (half-time feel)

Time signature — 4/4

Key characteristics — Distorted 808 cowbell loops, slowed vocal chops, saw-lead melodies, dark reverbed atmosphere

Era / region — Late 2010s–2020s. A Russia/Eastern European revival of 1990s Memphis phonk, amplified by TikTok drift edits.

Lyric vibe — Villain monologues, anti-hero swagger, minimal repeated hooks, often Russian, Japanese, or English phrases chopped beyond meaning. Lyrics serve atmosphere, not narrative.

Core prompt

```
phonk drift, dark menacing mood, distorted 808 cowbells, saw lead,
chopped vocal shouts, heavily reverbed, gritty tape compression, 142
BPM
```

Darker variant

```
phonk drift, evil horror atmosphere, detuned 808 cowbells, low
growling bass, whispered male vocal chops, heavy distortion, blown-out
mix, 138 BPM
```

Brighter variant

```
phonk drift, euphoric night driving mood, punchy cowbells, bright saw
lead, chopped female vocal hook, tape hiss, neon tunnel energy, 148
BPM
```

Common failure modes

- Cowbells come out too clean — add "distorted, saturated, gritty" to force the Memphis texture.
- Tempo drifts upward into hardstyle territory — lock the BPM explicitly and add "half-time feel".
- Vocal chops sound too melodic — specify "shouted, chopped, repeated phrase" to kill verse structure.

The move — Phonk drift is the only genre where you should actively request a worse mix. "Blown-out," "over-compressed," and "clipping" are features, not bugs. The genre was born from sampled cassette warble, and a pristine master sounds wrong.

Phonk (Memphis)

TL;DR — Raw, lo-fi, tape-warbled hip-hop from 1990s Memphis, the origin point of the entire phonk family.

BPM — 65–80 BPM (straight, not half-time)

Time signature — 4/4

Key characteristics — Sampled soul and organ loops, TR-808 drums, muffled rap vocals, cassette hiss baked in

Era / region — Early–mid 1990s Memphis, Tennessee; tape-trade distribution, cult status

Lyric vibe — Street-level, occult-tinged, conspiratorial. Drawled delivery, heavy Southern slang, grim humor. Hooks repeat like incantations.

Core prompt

```
Memphis phonk, raw lo-fi mood, sampled soul organ loop, classic 808 drum machine, muffled male rap vocals, cassette hiss, tape warble, 72 BPM
```

Darker variant

```
Memphis phonk, occult horror mood, minor key organ sample, booming 808 kick, whispered menacing male vocal, heavy tape saturation, vinyl pops, 68 BPM
```

Brighter variant

```
Memphis phonk, nostalgic summer mood, warm soul sample, laid-back 808 groove, smooth male rap flow, tape hiss, faded vinyl texture, 78 BPM
```

Common failure modes

- Output sounds like modern trap — stress "1990s" and "cassette" aggressively, and keep BPM under 80.
- Vocals come out too clean — add "muffled, tape-warbled, low-fidelity".
- Drums sound programmed and tight — ask for "loose, swung, classic 808" to get the dusty feel.

CHAPTER 06 · FIX WORKFLOWS

Troubleshooting — 18 Failure Modes

The chapter you'll dog-ear

Every Suno creator hits the same walls in their first fifty generations. Not random walls — specific, recognizable, repeating walls. The output comes out mumbled, or the chorus collapses, or the bridge hijacks a song that was working, or the vocal suddenly sounds like a different person halfway through. These aren't bad luck. They're failure modes, and almost all of them have a fix you can apply before you waste another credit.

This chapter catalogs the sixteen most common failure modes, in roughly the order new users encounter them. For each, you get: a description of what the output actually sounds like when this fails (so you can identify it), the underlying reason it happens, a concrete fix workflow you can run, a Before/After prompt pair showing the fix in practice, and an escalation path for when the fix doesn't take. If a particular section feels like it's describing a song you just generated, good — that's the point. Flip to it, run the workflow, re-generate.

A note on evidence. Suno's public documentation is thin, and a lot of what follows is community-tested rather than officially confirmed. Where a claim is derived from hands-on testing rather than documented behavior, I say so.

And one blanket caveat that covers the whole chapter, because it would be tedious to repeat it on every page. Several fixes below explain *why* something happens by reasoning about the model — what tends to sit near what, what a genre looks like in the material these systems learn from. **Treat every one of those explanations as our best working theory, never as fact.** Suno publishes nothing about its architecture or its training data, so nobody outside the company can verify a mechanism, us included. What we can stand behind is the pattern and the fix: this input reliably produces that problem, and this change reliably improves it. That part is testable, and it is what you are actually buying. If a fix works, it works whether or not our explanation of it is right. Everything else is what you'd find in Suno's help center or reproducible in five minutes of experimentation.

One principle before the catalog: edit, don't re-roll. The instinct when a generation comes back wrong is to hit generate again and hope for better luck. **Before any-**

thing below: Studio Mode needs a paid plan. It is not available on Free. Pro gives you a simplified Song Editor; the full timeline — and, since August 2026, MIDI, the effects chain and automation — is Premier. Roughly half the fixes in this chapter escalate to "take it into Studio," so if you are on Free, that escalation is not available to you yet and regenerating with a corrected prompt is your lever. Every fix below still starts with a prompt-side fix for exactly that reason.

Resist it. Suno's Studio Mode gives you surgical tools — **Replace Section** (regenerate just the bad bars, with a prompt), **Extend** (add length without re-doing the whole track), **Crop** (trim a bad intro or a glitched tail), and **Split from Mix** (separate stems to rebalance levels) — and almost every problem in this chapter has a fix that preserves the parts you already liked. Re-rolling throws away a good verse to fix a bad bridge and costs you credits for the privilege. When a fix below says "regenerate," it means regenerate *the section*, not the song, unless the whole prompt was wrong. The Studio edit tools are covered in depth in the v6 features chapter; this chapter assumes you'll reach for them first.

And four anti-patterns to rule out first. Before you diagnose anything below, check that your prompt isn't doing one of these — they cause a disproportionate share of bad generations, and creators report the same four over and over:

- **Over-stuffing the style field.** Past roughly seven tags (or about forty words), the descriptors start fighting each other and the output turns to mud. More tags is not more control; it's more conflict. Trim to the few that actually define the sound.
- **Conflicting tags.** "Aggressive *and* peaceful," "lo-fi *and* studio quality," "minimal *and* wall of sound" — contradictions force the model to average them into something that's neither. Pick a direction.
- **Relying on placebo parameter tags.** Percentage and numeric tags dropped into the lyrics field — `[Reverb: 30%]`, `[Bass: 80%]`, `[BPM: 120]` — don't do what they look like they do; Suno doesn't parse them as parameters. Describe the sound in words instead, and put BPM and key in the style box. (The full truth on this is in the tag reference chapter.)
- **Too many exclusions.** More than about five items in the Exclude field tends to thin the whole output, not just remove the unwanted parts. Prioritize the two or three exclusions that matter most.

Let's go.

1. Vocals Come Out Mumbled or Slurred

The symptom — The vocal is in the song, the melody is clear, the production is fine, but you can't actually make out what the singer is saying. Consonants get swallowed. Word boundaries blur. You catch maybe one line in four, and the rest is a kind of musical mouth-noise that sounds like a singer performing in a language you don't quite speak. It's especially common on ballads, R&B, and anything with heavy reverb on the vocal.

Why it happens — The pattern creators consistently observe is that the melodic line wins and the consonants bend around it — we can only describe the behaviour, not the cause, because Suno publishes nothing about how the vocal model works. When your lyric has too many consonant clusters packed into a short melodic phrase, or when the syllable density exceeds what the genre's native breathing pattern can accommodate, the model compresses the harder sounds into mush to keep the melodic line intact. Heavy reverb in the production prompt makes it worse — the model knows the consonants are going to get washed anyway, so it under-articulates them to start with.

The fix workflow

1. Count the syllables per line in your lyric. If any line in a chorus exceeds 10 syllables, rewrite it down to 8.
2. Break up consonant clusters. "Strength" and "twelfth" and "crunched" are phonetic landmines — replace them with softer equivalents.
3. Remove any "reverb-heavy", "washed out", or "cathedral" tags from the production prompt. Test dry first, then add reverb back only if the track still works.
4. Add `clear diction` or `articulate delivery` to the vocal tag block.
5. Re-generate and listen specifically for the first line of the chorus. If that's clear, the rest usually is too.

A note on *levels* versus *diction*. If the vocal is articulate but simply too quiet, too hot, or buried under the drums, that's a mix-balance problem, not a phonetics problem — and the fix is not a re-roll. Take the track you already have into Studio Mode

and use **Split from Mix** (or **Advanced Split**) to separate the stems, then rebalance the vocal against the instrumental. Re-rolling gambles the whole performance to fix a level; rebalancing fixes the level and keeps the take. The stem tools are covered in depth in the v6 features chapter — here, just remember: mud is a stems job, not a re-generation job.

Before

```
dream pop, ethereal, washed out, heavy cathedral reverb, breathy
female vocal, slow
```

After

```
dream pop, atmospheric, clear articulate female vocal, moderate plate
reverb, 85 BPM
```

When the fix doesn't work — If the vocal is still mumbled after simplifying the lyric and drying out the production, the problem is almost certainly the genre tag itself. "Shoegaze" and "dream pop" have diction blur baked into the training data. Switch to a cleaner adjacent genre ("indie pop", "slowcore", "chamber pop") and the vocal will sharpen up immediately, at which point you can layer atmospheric production back in through tags like "ambient pads" or "reverb tails" without sacrificing clarity.

---

2. Output Sounds Generic and Anonymous

The symptom — The track is competent. The mix is clean. Nothing is broken. But it could have been generated by anyone, for any purpose, and it doesn't feel like it belongs to a specific artist or project. It's the musical equivalent of stock photography. You listen back the next day and can't remember which generation was yours and which was the demo track Suno shows on its homepage.

Why it happens — Your prompt is describing a category, not a character. "Indie folk, acoustic, male vocal" is four genre-adjacent tags that could describe any one of ten thousand songs. Suno's default behavior when given a vague prompt is to

regress to the most statistically average interpretation — the median of its training data for that category. The median is, by definition, generic.

The fix workflow

1. Add a vocal character descriptor: register, texture, phrasing behavior. "Weathered male baritone" is worth more than "male vocal."
2. Add an era or production aesthetic: "early 70s analog warmth", "mid-80s reverb plate", "modern lo-fi bedroom recording."
3. Add one unexpected instrument that isn't typical for the genre. For indie folk, that might be "upright bass" or "pedal steel" or "dusty Rhodes piano."
4. Add a mood tag that isn't interchangeable with "sad" or "happy." Try "rueful", "restless", "unsettled dawn", "late-night kitchen."
5. If the result still feels generic, save your favorite candidate as a Voice and build forward from there. Voices lock in specificity across a run.

Before

```
indie folk, acoustic guitar, male vocal, melancholy
```

After

```
indie folk, weathered male baritone phrasing behind the beat,
fingerpicked acoustic, upright bass, dusty Rhodes piano, early 70s
analog warmth, rueful and restless, 92 BPM
```

When the fix doesn't work — When every generation still feels anonymous no matter how specific you get, the culprit is usually that you're chasing a genre Suno treats as a "safe" default. Folk, bedroom pop, and generic synthwave all have this problem. The escape hatch is to pull one tag from an adjacent subgenre that's more narrowly defined — "Appalachian", "Swedish pop", "Italo disco", "Celtic fusion" — which forces the model out of its median interpretation and into a more specific pocket. From there, your descriptive tags have something to attach to.