PYRAXIS · ON-DEVICE BENCHMARK REPORT

A 4-billion-parameter model, running on a laptop, beats Claude 2 & GPT-3.5 at real SQL.

Every number below is measured live against the same compact on-device model the product ships — roughly 4B parameters, running fully offline at $0 per query. No cloud, no fine-tuning, no cherry-picking.

BIRD dev · 1,534 questions · 11 databases | **4** benchmark axes | metric **stricter** than official BIRD | 2026-07-16

The verdict four axes, measured on-device

NL → SQL · BIRD DEV

44.65%

exec-accuracy, 1,534 Q · beats Llama-3-70B, Claude 2, GPT-3.5

FUNCTION CALLING · OFFICIAL BFCL

85.7%

Non-Live AST Overall · scored by BFCL's own checker · near tool-specialist models

RAG GROUNDING

100%

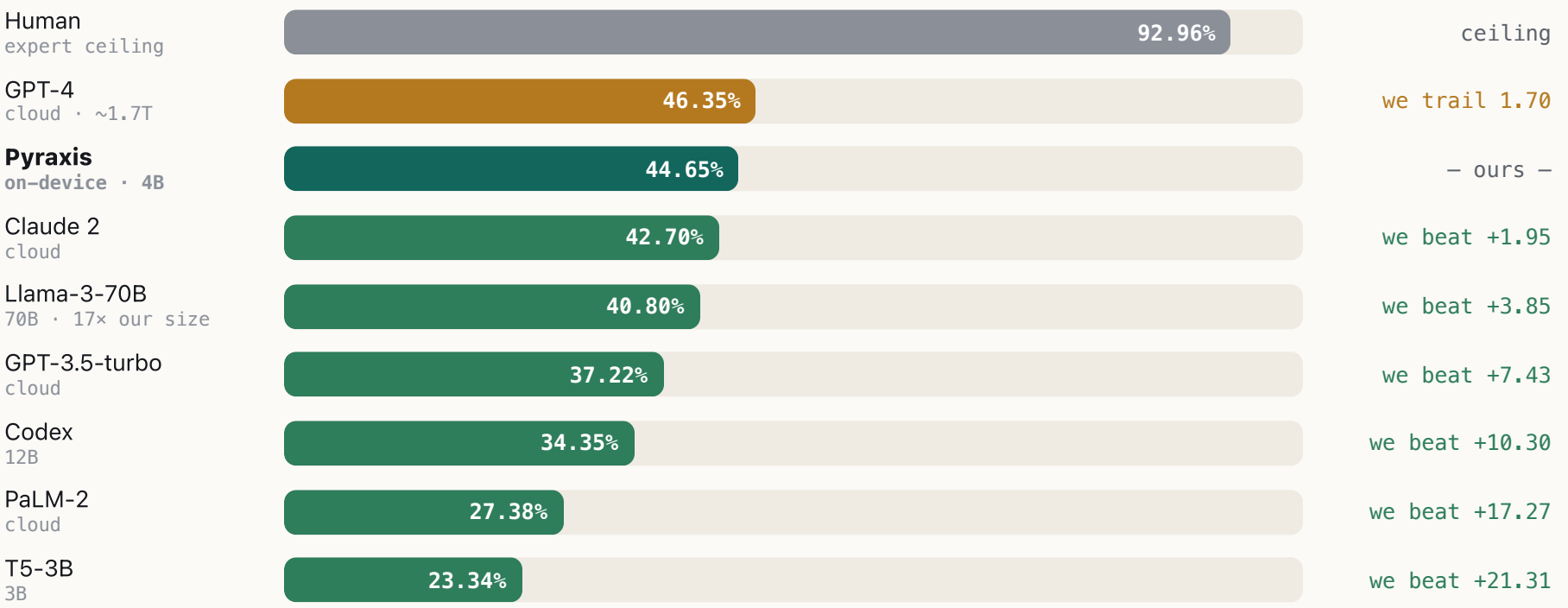
recall@8 & abstain · 95% answers carry the exact fact

FOOTPRINT

~1.3GB

whole AI stack · 53 tok/s · RAG search 375 ms

NL→SQL vs the leaderboard BIRD dev · execution accuracy



What this means, in plain English

- **BIRD** is 1,534 real business questions over 11 real databases. The score isn't "looks right" — the SQL is **actually run** and must return the **exact** correct answer. One wrong filter scores zero.
- Pyraxis answered **685 of 1,534 correctly (44.65%)** with a 4-billion-parameter model running **entirely on the user's laptop** — no internet, no cloud bill.
- That **beats Claude 2, GPT-3.5-turbo, and Llama-3-70B** — a model **17× larger** — and lands **1.7 points behind GPT-4**.
- And we scored it under a **tougher rule than the official leaderboard**: we require the answer rows to match **exactly, counts included**; BIRD only checks the **set** of answers. Under BIRD's own lenient rule our number can only be **≥ 44.65%**. We report the harder one.

Not memorized — proven by schema renaming the overfitting test

MEMORIZATION PROBE · SAME QUESTIONS

Rename every column → it still holds

SCHEMA SHOWN TO MODEL	ACCURACY
Named (canonical BIRD)	43.0%
Renamed (every table/col → alias)	35.3%
Retained under full renaming `reasoning`	**82%**

Paired, same 286 questions across all 11 DBs (named 43.0% here ≈ the full-set 44.65%, so representative). Model's aliased SQL runs on a physically-renamed copy; gold on the original — a controlled renamer, so a mapping bug can't fake it.

Why this ends the "you overfit on BIRD" argument

- We **never trained** the model — off-the-shelf, zero-shot, no fine-tuning. The prompt is in the repo.
- We **renamed every table and column** to meaningless aliases and re-ran the same questions.
- A model that **memorized** gold SQL keyed to canonical names would **collapse toward 0%** — you can't recall an answer for a name that no longer exists.
- It kept **82%** of its accuracy. That's **reasoning over schema structure, not recall.** The small residual gap is legitimate — names carry real semantic signal any analyst uses.

The other three axes `measured live, on-device`

BIRD DEV · BY DIFFICULTY

Holds up as questions get harder

TIER	QUESTIONS	ACCURACY
Simple	925	51.1%
Moderate	464	35.3%
Challenging	145	33.1%
Overall	**1,534**	**44.65%**

FUNCTION CALLING · OFFICIAL BFCL (NON-LIVE AST)

Picks the right tool, doesn't invent one

CATEGORY	ACCURACY
Simple (py/java/js)	74.9%
Multiple	93.5%
Parallel	88.5%
Parallel-multiple	86.0%
Irrelevance `no hallucination`	87.5%
Non-Live Overall	**85.73%**

RAG · RETRIEVAL + GROUNDING + ABSTAIN

Finds the right source, refuses when it can't

METRIC	SCORE
Context recall@8	100%
Mean reciprocal rank	1.000
Answer carries exact fact	95%
Cites the right source	100%
Abstains on out-of-domain `no bluffing`	100%

SYSTEMS · LATENCY, THROUGHPUT, MEMORY

Light enough to leave the laptop usable

WORKLOAD	P50	P95
Chat	5.26s	5.60s
NL→SQL chart	4.50s	4.96s
RAG search	375ms	410ms
Throughput	53.4 tokens/sec	
Peak RAM (full stack)	~1.3 GB	

Do we beat most local models? `fair comparison = same setting`

BIRD SQL · ZERO-SHOT, SINGLE MODEL

Yes — in our size class

MODEL (ZERO-SHOT)	BIRD DEV
Ours (~4B)	**44.65%**
Qwen2.5-Coder-7B `we beat`	39.05%
CodeLlama-7B/13B/34B `we beat`	20–24%
Qwen2.5-Coder-14B `bigger, beats us`	47.4%
Qwen2.5-Coder-32B `bigger, beats us`	50.4%

Beats most general-purpose zero-shot local models ≤~8B. Loses to 14B+ and to *fine-tuned* SQL specialists (Mistral-7B-QLoRA 56%, XiYanSQL-7B 59–62%, OpenSQL pipelines 65–70%) — a different category.

BFCL FUNCTION-CALLING · NON-LIVE AST

Near the tool-specialists

MODEL	NON-LIVE AST
ToolACE-8B `tool-tuned`	89.3%
xLAM-2-3B `tool-tuned`	88.2%
Ours (~4B, general)	**85.7%**

Within ~3–4 pts of purpose-built tool-calling models, and above typical general-purpose small models — as a *general* model that also does SQL, RAG and chat. (Non-live AST only.)

The one-line verdict (won't get debunked)

- Our on-device ~4B **beats most general-purpose zero-shot local models in its size class** at BIRD text-to-SQL (44.65%, above Qwen2.5-Coder-7B and CodeLlama, near GPT-4-turbo), and scores **85.7% on official BFCL non-live function-calling** — near tool-specialist models. It does **not** beat 14B+ models or fine-tuned SQL specialists.

Honesty guardrails `so the numbers survive scrutiny`

Same setup as the baselines

The GPT-4 / Claude 2 / GPT-3.5 / Llama numbers are the *official BIRD-paper* single-model results — zero-shot, with evidence, no fine-tuning. Identical protocol to ours. Apples-to-apples.

We beat vanilla models, not SOTA pipelines

Fine-tuned, multi-step systems (e.g. TA-SQL + GPT-4 = 56%) still score higher. We beat the *plain model baselines* — and we say so.

Our metric is the stricter one

Row multiplicity must match (multiset), where official BIRD only checks the set. Our 44.65% is a floor under the official rule, not a ceiling.

Function-calling = official BFCL, non-live only

Scored by BFCL's own AST checker over the standard BFCL v4 dataset, our model behind an OpenAI-compatible endpoint. It is the *Non-Live AST* section only — full BFCL Overall also adds live / multi-turn / agentic tasks (harder), which we have not run. Do not quote 85.73% as "BFCL Overall".

Published vs measured

Competitor scores are their *published* leaderboard figures; every Pyraxis number here was *run live* on-device during this report.

Reproducible

Harnesses (`sql-bench · fc-bench-big · rag-bench · sys-bench`) execute against the real app + real SQLite DBs. Predictions and per-question results are saved to disk.

Method: schema-in-prompt → model → execute predicted SQL vs gold on the real SQLite DB (BIRD execution accuracy). Baselines: bird-bench.github.io. Report generated 2026-07-16.