Unten bekommst du eine präzise Erklärung des Schemas und der Designentscheidungen, damit du es sicher begründen und erweitern kannst.

1) Zielbild und Prinzipien

- **Funktionsziel:** KFC-ähnliche Filtersuche über offizielle 5e-Statblocks, Benutzer-Login, Parties, Encounters; „mix & match“ (M:N) zwischen Parties und Encounters; Encounters bestehen aus beliebigen Statblocks mit Stückzahl.
- **Trennung Konzept vs. Druck:** Ein **creature** ist das konzeptionelle Wesen (z. B. „Goblin“). Der **vollständige Statblock** hängt an **creature_statblock** und ist **quellengebunden** (source_id) inkl. optionalem **variant_name** und **printed_name**. So deckst du ab: gleicher Kreaturenname, mehrere Statblocks in einer Quelle, abweichende Druckbezeichnungen, Seitenangaben.
- **MVP-Scope:** Keine Actions/Traits/Legendary Actions. „Legendary?“ nur als Flag zum Filtern.
- **Integrität & Erweiterbarkeit:** Starke Normalisierung (3NF), gepflegte Lookups mit RESTRICT, Detailtabellen mit CASCADE, aussagekräftige Checks, CI-Unique-Constraints, Indizes für KFC-Filter.

2) Bausteine im Detail

2.1 Extensions und Transaktion

- pgcrypto für UUID-Generierung in der DB (robuste, verteilte IDs; keine Kollisionen, keine Leaks von Insert-Zählungen).
- BEGIN/COMMIT um das Schema atomar zu erstellen.

2.2 Lookups (kuratiert, klein, ohne Auditing)

Warum:

- Referentielle Integrität, validierbare Werte, stabile Codes für UI/Filter.
- SMALLSERIAL reicht (kleine Kardinalität, schnelle Joins).

Tabellen:
ac_type, size, creature_type, alignment, damage_type, condition_type, speed_type, sense_type, language, environment, tag, skill, ability, source.

- code = maschinenlesbarer Schlüssel (stabil), name = Anzeige.
- source als Lookup, weil im MVP nur offizielle Quellen gepflegt werden.
- **cr_to_xp**: Lookup statt xp-Spalte. Vorteil: 1 Source of Truth, keine Inkonsistenzen, sauber 3NF; Performance ist trivial (kleine Tabelle, Index auf PK).

2.3 Benutzerverwaltung

app_user

- email CI-unique via (lower(email)).
- deleted_at = Soft-Delete-Marker; 30-Tage-Purge erfolgt **anwendungsseitig**.
- Encounters/Parties referenzieren owner_id ON DELETE CASCADE, d. h. beim späteren Hard-Delete des Users werden seine Daten sauber aufgeräumt.

2.4 Domänenmodell „Creature ↔ Statblock“

creature

- Nur der **konzeptionelle Name**. CI-Unique auf lower(name) vermeidet Duplikate („Goblin“ vs. „goblin“).
- Auditing-Felder (created_/updated_ at/by).

creature_statblock

- Volle **Statline** inkl. Quelle, **optionalem** variant_name (z. B. „Scout“), printed_name (exakte Druckbezeichnung), page_number, location_note.
- **Unique-Schlüssel:** (creature_id, source_id, lower(coalesce(variant_name,''))). Dadurch: mehrere Varianten pro Quelle, aber jede Variante eindeutig.
- **Attribute:**
    - Größe/Typ/Ausrichtung via Lookups.
    - AC: armor_class + ac_type_id + Freitext ac_notes für Nuancen („with shield“, „natural armor“).
    - HP: hit_points_avg + **gedruckte** hit_dice als Text (Originalformat beibehalten; Parsing optional später).
    - Ability Scores mit CHECK 1..30 (5e-Range).
    - passive_perception separat, da oft explizit angegeben.
    - challenge_rating NUMERIC(4,3) CHECK >= 0 (0, 1/8, 1/4, 1/2, 1..30).
    - is_legendary als boolesches Filterfeld (ohne Legendary-Actions im MVP).
    - Vollständiges Auditing.

Warum Statblocks an der Quelle:

- Offizielle Bücher drucken teils **abweichende oder spezialisierte** Statblocks für „dieselbe“ Kreatur.

- Mit variant_name deckst du Varianten **in derselben** Quelle robust ab.
- printed_name erlaubt 1:1-Referenz zur Veröffentlichung (Textsuche, Import-Matching).

2.5 Detailtabellen zum Statblock

Warum aufgesplittet:

- Mehrfachwerte (1:n) und Listen (Senses, Speeds, Immunities, …) werden normalisiert; so sind Abfragen und Filter performant, semantisch sauber und duplizierfrei.
- statblock_to_speed: je speed_type eine Zeile, can_hover für Fly-Spezialfall; PK schützt vor Duplikaten (ein Speed-Typ nur einmal pro Statblock).
- creature_statblock_save und creature_statblock_skill: **explizite Boni** statt Ableitung. Begründung: Offizielle Statblöcke enthalten oft Abweichungen (Expertise, runde/ungerade Werte, Sonderlogik); explizit speichern verhindert Fehlschlüsse.
- statblock_to_sense (mit Reichweite), statblock_to_language (Telepathie-Reichweite optional + Note).
- statblock_to_damage_vulnerability/_resistance/_immunity: drei Tabellen statt „Typ-Spalte“ in einer, so bleiben **FK-Beziehungen und semantische Indizes simpel**. qualifier als Freitext hält Feinheiten wie „from nonmagical attacks“.
- statblock_to_condition_immunity: bedingungsbasierte Immunitäten.
- statblock_to_environment, statblock_to_tag: KFC-Filterdimensionen, kuratiert.

2.6 Parties & Encounters

- **party**: gehört einem owner_id, hat name und ein globales party_level (MVP-einfach).
- **encounter**: gehört einem owner_id, frei benennbar, optional description.
- **M:N „mix & match“**: party_to_encounter (party_id, encounter_id) als PK-Join.
- **Begegnungsinhalt**: encounter_to_creature_statblock mit quantity und note.
    - **Warum eigener Surrogate-PK (BIGSERIAL) statt Unique(encounter_id, statblock_id)?**
      Flexibler: du kannst dasselbe Statblock **mehrfach** mit unterschiedlicher Zweck-Notiz hinzufügen (z. B. „archers on ridge“, „camp guards“), **oder** du nutzt eine Zeile mit quantity. Das Schema lässt beides zu. Wenn du

strikt Duplikate verhindern willst, ergänze später UNIQUE (encounter_id, creature_statblock_id).

2.7 Views

- v_creature_statblock_with_xp: bequemer Join statblock ↔ cr_to_xp, damit Queries XP direkt bekommen, ohne an die Lookup-Tabelle denken zu müssen.

2.8 Indizes

- **Suche:** lower(name)-Indizes auf creature.name, creature_statblock.printed_name, variant_name für CI-Suche.
- **Filter:** Einzel-Indizes auf CR, Size, Type, Alignment, Source, is_legendary.
- **Join-Hilfen:** Indizes auf allen 1:n-FK-Spalten (Senses, Speeds, …) und Encounter-Komposition.
- **Optional später:** pg_trgm für fuzzy Name-Suche; unaccent für diakritische Robustheit.

2.9 Constraints & Löschregeln

- **Lookups ON DELETE RESTRICT**: schützt vor versehentlichem Löschen kuratierter Werte.
- **Detailtabellen ON DELETE CASCADE**: löscht bei Statblock-Löschung alle zugehörigen Details sauber mit.
- **encounter_to_creature_statblock.creature_statblock_id ON DELETE RESTRICT**: Statblock kann nicht gelöscht werden, wenn er in einem gespeicherten Encounter verwendet wird (Datenerhalt).
- **Checks** sichern Datenqualität (z. B. challenge_rating >= 0, Ability 1..30, speed_ft >= 0, party_level 1..20).

3) Trade-offs und Alternativen

- **XP speichern vs. ableiten:** Ableiten über cr_to_xp ist konsistenter; wenn du Performance-Messungen brauchst, leg zusätzlich eine **materialisierte View** über v_creature_statblock_with_xp an.
- *Eine Tabelle für damage_ statt drei:** Möglich, aber Queries/Indizes werden unübersichtlicher (brauchen dann Filterspalte + Teilindizes). Drei Tabellen sind klarer und performanter für Filter.
- **creature-Name CI-unique:** Praktisch, aber bei echten Kollisionen (gleichnamige Konzepte) müsstest du per Suffix differenzieren; im offiziellen 5e-Kanon i. d. R. unkritisch.

- **encounter_to_creature_statblock Unique:** Wenn du keine getrennten Notizzwecke brauchst, kannst du das Duplikat-Risiko per UNIQUE (encounter_id, creature_statblock_id) eliminieren.

```
--
================================================================================

-- Encounterplanner (MVP) – PostgreSQL Schema

--
================================================================================


-- UUIDs via pgcrypto

CREATE EXTENSION IF NOT EXISTS pgcrypto;


BEGIN;


--
================================================================================

-- Lookup Tables (curated, small cardinality, no auditing)

--
================================================================================


CREATE TABLE ac_type (

  id          SMALLSERIAL PRIMARY KEY,

  code        TEXT NOT NULL UNIQUE,       -- e.g. 'natural_armor', 'armor', 'unarmored', 'mage_armor'
```

```
  name        TEXT NOT NULL
);

CREATE TABLE size (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,      -- 'tiny','small','medium','large','huge','gargantuan'
  name        TEXT NOT NULL
);

CREATE TABLE creature_type (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,      -- 'aberration','beast','celestial', ...
  name        TEXT NOT NULL
);

CREATE TABLE alignment (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,      -- 'unaligned','lg','ng','cg','ln','nn','cn','le','ne','ce'
  name        TEXT NOT NULL
);

CREATE TABLE damage_type (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,      -- 'acid','bludgeoning','cold','fire','force','lightning','necrotic','piercing','poison','psychic','radiant','slashing','thunder'
  name        TEXT NOT NULL
);
```

```
CREATE TABLE condition_type (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,      -- 'blinded','charmed','deafened','frightened',
...
  name        TEXT NOT NULL
);

CREATE TABLE speed_type (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,      -- 'walk','fly','swim','burrow','climb'
  name        TEXT NOT NULL
);

CREATE TABLE sense_type (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,      --
'darkvision','blindsight','tremorsense','truesight'
  name        TEXT NOT NULL
);

CREATE TABLE language (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,      -- 'common','draconic','telepathy', ...
  name        TEXT NOT NULL
);

CREATE TABLE environment (
  id          SMALLSERIAL PRIMARY KEY,
```

```
  code        TEXT NOT NULL UNIQUE,       -- KFC-like: 'arctic','coast','desert','forest','grassland','hill','mountain','swamp','underdark','urban', ...
  name        TEXT NOT NULL
);

CREATE TABLE tag (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,       -- curated tags used for filtering
  name        TEXT NOT NULL
);

-- Skills (explicit bonuses stored per statblock)
CREATE TABLE skill (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,       -- 'acrobatics','animal_handling','arcana','athletics','deception','history','insight','intimidation','investigation','medicine','nature','perception','performance','persuasion','religion','sleight_of_hand','stealth','survival'
  name        TEXT NOT NULL
);

-- Ability lookup for saving throws (keine Enums für bessere Portabilität)
CREATE TABLE ability (
  id          SMALLSERIAL PRIMARY KEY,
  code        TEXT NOT NULL UNIQUE,       -- 'str','dex','con','int','wis','cha'
  name        TEXT NOT NULL
);

-- Official sources (curated -> treat as lookup)
```

```
CREATE TABLE source (
  id           SMALLSERIAL PRIMARY KEY,
  code         TEXT NOT NULL UNIQUE,       -- 'MM2014','VGM','MToF', ...
  title        TEXT NOT NULL,
  abbrev       TEXT,
  release_year   SMALLINT,
  publisher     TEXT
);

-- Challenge Rating -> XP mapping (derived at query time; no xp column on statblock)
CREATE TABLE cr_to_xp (
  challenge_rating   NUMERIC(4,3) PRIMARY KEY, -- supports 0, 0.125, 0.25, 0.5, 1...30
  xp             INTEGER NOT NULL CHECK (xp >= 0)
);


--
================================================================================
------
-- Users (Soft-Delete only here)
--
================================================================================
------

CREATE TABLE app_user (
  id           UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  email        TEXT NOT NULL,
  password_hash   TEXT NOT NULL,
  created_at     TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_at     TIMESTAMPTZ NOT NULL DEFAULT now(),
```

```
  deleted_at     TIMESTAMPTZ,              -- soft-delete marker
  CONSTRAINT uq_app_user_email_ci UNIQUE ((lower(email)))
);

--
================================================================
======
-- Core Domain Entities (with auditing)
--
================================================================
======

-- Conceptual creature (no stats here; stats live at the source-specific statblock)
CREATE TABLE creature (
  id           UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  name         TEXT NOT NULL,
  -- optional metadata for future: slug, description, etc.
  created_at     TIMESTAMPTZ NOT NULL DEFAULT now(),
  created_by     UUID REFERENCES app_user(id) ON DELETE SET NULL,
  updated_at     TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_by     UUID REFERENCES app_user(id) ON DELETE SET NULL,
  CONSTRAINT uq_creature_name_ci UNIQUE ((lower(name)))
);

-- Source-bound statblock (stores the full printed statline)
CREATE TABLE creature_statblock (
  id             UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  creature_id       UUID NOT NULL REFERENCES creature(id) ON DELETE CASCADE,
  source_id         SMALLINT NOT NULL REFERENCES source(id) ON DELETE RESTRICT,
```

```
  -- variant handling within same source (nullable)
  variant_name      TEXT,              -- e.g. 'Tribal', 'Scout'
  printed_name      TEXT,              -- exact printed name in source (may differ)
  page_number       SMALLINT,          -- optional
  location_note     TEXT,              -- e.g. 'Appendix A'

  size_id           SMALLINT NOT NULL REFERENCES size(id) ON DELETE RESTRICT,
  type_id           SMALLINT NOT NULL REFERENCES creature_type(id) ON DELETE
RESTRICT,
  alignment_id      SMALLINT NOT NULL REFERENCES alignment(id) ON DELETE
RESTRICT,

  armor_class       SMALLINT NOT NULL CHECK (armor_class >= 0),
  ac_type_id        SMALLINT REFERENCES ac_type(id) ON DELETE RESTRICT,
  ac_notes          TEXT,              -- free-form qualifier, e.g. "natural armor", "with shield"

  hit_points_avg    INTEGER NOT NULL CHECK (hit_points_avg >= 0),
  hit_dice          TEXT NOT NULL,     -- e.g. "4d8+4"

  speed_notes       TEXT,              -- optional general speed notes

  strength          SMALLINT NOT NULL CHECK (strength BETWEEN 1 AND 30),
  dexterity         SMALLINT NOT NULL CHECK (dexterity BETWEEN 1 AND 30),
  constitution      SMALLINT NOT NULL CHECK (constitution BETWEEN 1 AND 30),
  intelligence      SMALLINT NOT NULL CHECK (intelligence BETWEEN 1 AND 30),
  wisdom            SMALLINT NOT NULL CHECK (wisdom BETWEEN 1 AND 30),
  charisma          SMALLINT NOT NULL CHECK (charisma BETWEEN 1 AND 30),
```

```
  senses_notes       TEXT,                -- optional general senses notes
  passive_perception  SMALLINT,           -- printed passive perception

  languages_notes    TEXT,                -- optional general languages notes

  challenge_rating   NUMERIC(4,3) NOT NULL CHECK (challenge_rating >= 0),
  is_legendary       BOOLEAN NOT NULL DEFAULT FALSE,  -- for filtering; no legendary actions modeled

  created_at         TIMESTAMPTZ NOT NULL DEFAULT now(),
  created_by         UUID REFERENCES app_user(id) ON DELETE SET NULL,
  updated_at         TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_by         UUID REFERENCES app_user(id) ON DELETE SET NULL,

  -- unique per (creature, source, variant)
  CONSTRAINT uq_statblock_variant UNIQUE (creature_id, source_id, (lower(COALESCE(variant_name, ''))))
);

-- Speeds (multiple types per statblock)
CREATE TABLE statblock_to_speed (
  creature_statblock_id   UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE CASCADE,
  speed_type_id          SMALLINT NOT NULL REFERENCES speed_type(id) ON DELETE RESTRICT,
  speed_ft               SMALLINT NOT NULL CHECK (speed_ft >= 0),
  can_hover              BOOLEAN NOT NULL DEFAULT FALSE, -- relevant for fly
  PRIMARY KEY (creature_statblock_id, speed_type_id)
);
```

```
-- Saving throws (explicit bonuses)
CREATE TABLE creature_statblock_save (
  creature_statblock_id  UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE CASCADE,
  ability_id          SMALLINT NOT NULL REFERENCES ability(id) ON DELETE RESTRICT,
  bonus               SMALLINT NOT NULL,
  PRIMARY KEY (creature_statblock_id, ability_id)
);

-- Skills (explicit bonuses)
CREATE TABLE creature_statblock_skill (
  creature_statblock_id  UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE CASCADE,
  skill_id            SMALLINT NOT NULL REFERENCES skill(id) ON DELETE RESTRICT,
  bonus               SMALLINT NOT NULL,
  PRIMARY KEY (creature_statblock_id, skill_id)
);

-- Senses with ranges
CREATE TABLE statblock_to_sense (
  creature_statblock_id  UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE CASCADE,
  sense_type_id       SMALLINT NOT NULL REFERENCES sense_type(id) ON DELETE RESTRICT,
  range_ft            SMALLINT NOT NULL CHECK (range_ft >= 0),
  PRIMARY KEY (creature_statblock_id, sense_type_id)
);
```

```
-- Languages (telepathy range optional per record)
CREATE TABLE statblock_to_language (
  creature_statblock_id   UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE CASCADE,
  language_id          SMALLINT NOT NULL REFERENCES language(id) ON DELETE RESTRICT,
  telepathy_range_ft     SMALLINT CHECK (telepathy_range_ft >= 0),
  note                 TEXT,
  PRIMARY KEY (creature_statblock_id, language_id)
);

-- Damage vulnerabilities / resistances / immunities (with free-text qualifiers)
CREATE TABLE statblock_to_damage_vulnerability (
  creature_statblock_id   UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE CASCADE,
  damage_type_id       SMALLINT NOT NULL REFERENCES damage_type(id) ON DELETE RESTRICT,
  qualifier            TEXT,   -- e.g. "from nonmagical attacks"
  PRIMARY KEY (creature_statblock_id, damage_type_id)
);

CREATE TABLE statblock_to_damage_resistance (
  creature_statblock_id   UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE CASCADE,
  damage_type_id       SMALLINT NOT NULL REFERENCES damage_type(id) ON DELETE RESTRICT,
  qualifier            TEXT,
  PRIMARY KEY (creature_statblock_id, damage_type_id)
);
```

```
CREATE TABLE statblock_to_damage_immunity (
  creature_statblock_id   UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE CASCADE,
  damage_type_id        SMALLINT NOT NULL REFERENCES damage_type(id) ON DELETE RESTRICT,
  qualifier             TEXT,
  PRIMARY KEY (creature_statblock_id, damage_type_id)
);

-- Condition immunities
CREATE TABLE statblock_to_condition_immunity (
  creature_statblock_id   UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE CASCADE,
  condition_type_id      SMALLINT NOT NULL REFERENCES condition_type(id) ON DELETE RESTRICT,
  PRIMARY KEY (creature_statblock_id, condition_type_id)
);

-- Environments (KFC-like filtering)
CREATE TABLE statblock_to_environment (
  creature_statblock_id   UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE CASCADE,
  environment_id        SMALLINT NOT NULL REFERENCES environment(id) ON DELETE RESTRICT,
  PRIMARY KEY (creature_statblock_id, environment_id)
);

-- Curated tags
CREATE TABLE statblock_to_tag (
```

```
  creature_statblock_id   UUID NOT NULL REFERENCES creature_statblock(id) ON
DELETE CASCADE,
  tag_id              SMALLINT NOT NULL REFERENCES tag(id) ON DELETE RESTRICT,
  PRIMARY KEY (creature_statblock_id, tag_id)
);


--
================================================================================
------
-- Parties & Encounters (owned by users)
--
================================================================================
------


CREATE TABLE party (
  id          UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  owner_id      UUID NOT NULL REFERENCES app_user(id) ON DELETE CASCADE,
  name          TEXT NOT NULL,
  party_level    SMALLINT NOT NULL CHECK (party_level BETWEEN 1 AND 20),
  created_at     TIMESTAMPTZ NOT NULL DEFAULT now(),
  created_by     UUID REFERENCES app_user(id) ON DELETE SET NULL,
  updated_at     TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_by     UUID REFERENCES app_user(id) ON DELETE SET NULL
);


CREATE TABLE encounter (
  id          UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  owner_id      UUID NOT NULL REFERENCES app_user(id) ON DELETE CASCADE,
  name          TEXT NOT NULL,
```

```
  description    TEXT,
  created_at     TIMESTAMPTZ NOT NULL DEFAULT now(),
  created_by     UUID REFERENCES app_user(id) ON DELETE SET NULL,
  updated_at     TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_by     UUID REFERENCES app_user(id) ON DELETE SET NULL
);

-- M:N: Parties ↔ Encounters (mix & match)
CREATE TABLE party_to_encounter (
  party_id       UUID NOT NULL REFERENCES party(id) ON DELETE CASCADE,
  encounter_id   UUID NOT NULL REFERENCES encounter(id) ON DELETE CASCADE,
  PRIMARY KEY (party_id, encounter_id)
);

-- Encounter ↔ Statblocks (with quantity)
CREATE TABLE encounter_to_creature_statblock (
  id                 BIGSERIAL PRIMARY KEY,
  encounter_id       UUID NOT NULL REFERENCES encounter(id) ON DELETE CASCADE,
  creature_statblock_id  UUID NOT NULL REFERENCES creature_statblock(id) ON DELETE RESTRICT,
  quantity           INTEGER NOT NULL DEFAULT 1 CHECK (quantity >= 1),
  note               TEXT
);

-- ================================================================================

-- Views (optional convenience)
```

```
--
================================================================================
======

-- Derive XP for a statblock from CR via cr_to_xp
CREATE VIEW v_creature_statblock_with_xp AS
SELECT
  cs.*,
  ctx.xp
FROM creature_statblock cs
LEFT JOIN cr_to_xp ctx
 ON ctx.challenge_rating = cs.challenge_rating;

--
================================================================================
======
-- Indexes for common filters/search
--
================================================================================
======

-- name searches
CREATE INDEX idx_creature_name_ci ON creature (lower(name));
CREATE INDEX idx_statblock_printed_name_ci ON creature_statblock
(lower(printed_name));
CREATE INDEX idx_statblock_variant_name_ci ON creature_statblock
(lower(variant_name));

-- KFC-like filters
CREATE INDEX idx_statblock_cr ON creature_statblock (challenge_rating);
```

```
CREATE INDEX idx_statblock_size ON creature_statblock (size_id);

CREATE INDEX idx_statblock_type ON creature_statblock (type_id);

CREATE INDEX idx_statblock_alignment ON creature_statblock (alignment_id);

CREATE INDEX idx_statblock_source ON creature_statblock (source_id);

CREATE INDEX idx_statblock_is_legendary ON creature_statblock (is_legendary);


-- join helpers

CREATE INDEX idx_speed_statblock ON statblock_to_speed (creature_statblock_id);

CREATE INDEX idx_sense_statblock ON statblock_to_sense (creature_statblock_id);

CREATE INDEX idx_lang_statblock ON statblock_to_language (creature_statblock_id);

CREATE INDEX idx_vuln_statblock ON statblock_to_damage_vulnerability
(creature_statblock_id);

CREATE INDEX idx_res_statblock ON statblock_to_damage_resistance
(creature_statblock_id);

CREATE INDEX idx_imm_statblock ON statblock_to_damage_immunity
(creature_statblock_id);

CREATE INDEX idx_condimm_statblock ON statblock_to_condition_immunity
(creature_statblock_id);

CREATE INDEX idx_env_statblock ON statblock_to_environment
(creature_statblock_id);

CREATE INDEX idx_tag_statblock ON statblock_to_tag (creature_statblock_id);


-- encounter composition

CREATE INDEX idx_enc_to_cs_enc ON encounter_to_creature_statblock
(encounter_id);

CREATE INDEX idx_enc_to_cs_cs  ON encounter_to_creature_statblock
(creature_statblock_id);


COMMIT;
```