Query 1: Cards with Mid Price Above Average
π card_name, mid_price, set_name (
σ mid_price > γ AVG(mid_price) (price_tracker)
(price_tracker ⋈ card ⋈ card_set)
)

Query 2: Post-2020 Standard Legal Sets
π set_name, release_date (
σ release_date > '2020-01-01' ∧ legality_id ∈ π legality_id (σ standard=true (legality))
(card_set)
)

Query 3: Max Mid Price per Set
π set_name, card_name, mid_price (
card_set ⋈
(card ⋈ price_tracker) ⋈
γ set_id; MAX(mid_price)→max_price (price_tracker ⋈ card)
on set_id and mid_price = max_price
)

Query 4: Price Rank by Rarity
π card_name, rarity, mid_price, RANK() (
(card ⋈ price_tracker ⋈ rarity)
sort by rarity, mid_price DESC
)

RA Note: Window functions require sorting and partitioning which isn't native to RA. This is approximated with sorting and sequence numbers.

Query 5: Release Order in Sets
π card_name, set_name, release_date, ROW_NUMBER() (
(card ⋈ card_set)
partition by set_id sort by release_date
)

Query 6: Price Tiers (Quartiles)
π card_name, mid_price, NTILE(4) (
(card ⋈ price_tracker)
sort by mid_price DESC
)

Query 7: Expanded Legal Cards
π card_name, set_name, expanded (
σ expanded=true (legality ⋈ card_set ⋈ card)
)

Query 8: Average HP >50 by Type
π type, AVG(hp) (

γ type; AVG(hp)→avg_hp (
type ⋈ card_type ⋈ card
)
)
σ avg_hp > 50

Query 9: Artists in "Base" Sets
π artist (
σ set_name LIKE '%Base%' (card ⋈ card_set)
)
remove duplicates

Query 10: Rarity Counts in "Evolving Skies"
π rarity, COUNT(card_id) (
γ rarity (
σ set_name='Evolving Skies' (rarity ⋈ card ⋈ card_set)
)
)

Query 11: Rarity Price Statistics
γ rarity; AVG(market_price), COUNT(*), STDDEV_SAMP(market_price) (
rarity ⋈ card ⋈ price_tracker
)

Query 12: Cards with Subtypes
π card_id, card_name (
σ EXISTS(σ card_id=c.card_id (card_subtype)) (card)
)

Query 13: Cards in Base Set or Jungle
π card_name (
σ set_name='Base Set' (card ⋈ card_set)
) ∪
π card_name (
σ set_name='Jungle' (card ⋈ card_set)
)

Query 14: Price Variance per Set
γ set_name; VAR_SAMP(market_price) (
card_set ⋈ card ⋈ price_tracker
)

Query 15: Sets with Avg High Price >50
π set_name (
σ avg_high_price >50 (
γ set_name; AVG(high_price)→avg_high_price (
card_set ⋈ card ⋈ price_tracker
)

)
)

Query 16: Unlimited Legal but Not Standard
π set_name (
card_set ⋈
(σ unlimited=true (legality)) - (σ standard=true (legality))
)

Query 17: Dual-Type Fire/Water Cards
π card_id, card_name (
(σ type='Fire' (type ⋈ card_type ⋈ card)) ∩
(σ type='Water' (type ⋈ card_type ⋈ card))
)

Query 18: Subtype Count per Card
π card_id, card_name, COUNT(subtype_id) (
card ⟕ card_subtype
)

Query 19: Avg Market Price per Set
π set_name, avg_market_price (
card_set ⋈
(γ set_id; AVG(market_price)→avg_market_price (card ⋈ price_tracker))
)

Query 20: Above-Average Priced Cards
π card_id, card_name, market_price, set_name (
σ market_price > (
γ set_id; AVG(market_price) (card ⋈ price_tracker)
)
(card ⋈ price_tracker ⋈ card_set)
)

Note: Relational Algebra uses Greek symbols in formal notation (π=projection, σ=selection, ⋈=join, γ=aggregation). Subqueries are represented using nested operations.

Entities	Attributes	Constraints
Price_tracker Tracks pricing information for Pokémon cards.	**price_tracker_id**: BIGINT (Auto-incremented unique identifier). **url:** VARCHAR(200) (A string up to 200 characters, for the webpage URL). **card_id:** VARCHAR(50) (A string up to 50 characters, references the card table, must not be null). **updated_at:** DATE (The date the pricing information was last updated). **card_type:** VARCHAR(200) (Describes the card type). **low_price, mid_price, high_price, market_price:** DECIMAL(12, 2) (Stores monetary values with 2 decimal places).	**price_tracker_id** is the primary key. **card_id** references the card table and cannot be null.
Pokedex	**pokedex_id:** INT (A unique integer representing the Pokémon's Pokedex entry, must not be null). **pokedex_region:** VARCHAR(50) (A string representing the region of the pokedex entry, must not be null).	**pokedex_id** is the primary key.
Pokedex_card Links Pokémon cards to Pokémon from the Pokedex.	**card_id:** VARCHAR(50) (References the card table, must not be null). **pokedex_id:** INT (References the pokedex table, must not be null).	The combination of **card_id** and **pokedex_id** is the primary key (ensures each mapping is unique). **pokedex_id** references the subtype table and cannot be null. **card_id** references the card table and cannot be null.
Rarity Defines rarity levels for Pokémon cards.	**rarity_id:** INT (A unique identifier for each rarity). **rarity:** VARCHAR(50) (A string up to 50 characters, must be unique and not null).	**rarity_id** is the primary key. **rarity** must be unique and not null.
Subtype	**subtype_id:** INT (A unique identifier for each subtype).	**subtype_id** is the primary key.

Categorizes cards into specific subtypes (e.g., "EX", "VSTAR").	**subtype:** VARCHAR(100) (A unique string up to 100 characters, must not be null).	
Card_subtype Links Pokémon cards to their respective subtype.	**subtype_id:** INT (References the subtype table, must not be null). **card_id:** VARCHAR(50) (References the card table, must not be null).	The combination of **card_id** and **subtype_id** is the primary key (ensures each mapping is unique). **subtype_id** references the subtype table and cannot be null. **card_id** references the card table and cannot be null.
Supertype Groups cards into broader categories (e.g., "Pokémon", "Trainer").	**supertype_id:** INT (A unique identifier for each supertype). **supertype:** VARCHAR(50) (A string up to 50 characters, must be unique and not null).	**supertype_id** is the primary key. **supertype** must be unique and not null.
Type Defines elemental types for cards (e.g., "Fire", "Water").	**type_id:** INT (A unique identifier for each type). **type:** VARCHAR(50) (A string up to 50 characters, must be unique and not null).	**type_id** is the primary key. **type** must be unique and not null.
Card_type Links Pokémon cards to their respective elemental types.	**card_type_id:** BIGINT (Auto-incremented unique identifier for each record). **type_id:** INT (References the type table, must not be null). **card_id:** VARCHAR(50) (References the card table, must not be null).	**card_type_id** is the primary key. **type_id** references the type table and cannot be null. **card_id** references the card table and cannot be null.
Legality Tracks whether cards or sets are legal in specific play formats.	**legality_id:** BIGINT (Auto-incremented unique identifier). **unlimited:** BOOLEAN (Indicates legality in the Unlimited format, defaults to false). **standard:** BOOLEAN (Indicates legality in the Standard format, defaults to false). **expanded:** BOOLEAN (Indicates legality in the Expanded format, defaults to false).	**legality_id** is the primary key.

Card_set Organizes Pokémon cards into sets (e.g., "Base Set", "Sword & Shield").	**set_id:** VARCHAR(100) (Unique ID for the set). **set_name:** VARCHAR(150) (Name of the set, must be unique and not null). **legality_id:** BIGSERIAL (References the legality table). **series:** VARCHAR(150) (Name of the series the set belongs to). **printed_total:** INT (The number of cards printed in the set). **total:** INT (The total number of cards in the set). **ptcgo_code:** VARCHAR(50) (A unique code for the Pokémon TCG Online, optional). **release_date:** DATE (The release date of the set). **updated_at:** DATE (The date the set information was last updated). **symbol_img:** VARCHAR(200) (URL for the set symbol). **logo_img:** VARCHAR(200) (URL for the set logo).	**set_id** is the primary key. **set_name** and **ptcgo_code** must be unique.
Card Contains all details about individual Pokémon cards.	**card_id:** VARCHAR(50) (Unique identifier for the card). **set_id:** VARCHAR(100) (References the card_set table, must not be null). **rarity_id:** INT (References the rarity table). **supertype_id:** INT (References the supertype table). **card_name:** VARCHAR(100) (Name of the card, must not be null). **number:** VARCHAR(50) (Card number within its set, must be unique). **artist:** VARCHAR(100) (Name of the artist who illustrated the card).	**card_id** is the primary key. **set_id**, **rarity_id**, and **supertype_id** are foreign keys. **number** must be unique.

	small_img: VARCHAR(200) (URL for a small image of the card). **large_img:** VARCHAR(200) (URL for a large image of the card). **hp:** INT (The card's hit points). **flavor_text:** VARCHAR(500) (Optional descriptive text for the card).	

Relation	Cardinality	Participation Constraint
Price_tracker ↔ card Each entry in price_tracker represents pricing data for a specific card.	One-to-One Each card can have at most one entry in price_tracker (pricing information is unique to the card). Each price_tracker record must reference exactly one card.	**price_tracker.card_id: Mandatory** (Pricing must belong to a card). **card.card_id: Optional** (A card may not yet have pricing information).
pokedex ↔ card (via pokedex_card) A card can represent a Pokémon species in the Pokedex.	Many-to-Many Many cards can reference the same pokedex_id (e.g., multiple Pikachu cards). Each pokedex entry may reference more than one card.	**pokedex_card.card_id: Mandatory** (Every pokedex association must belong to a valid card). **pokedex_card.pokedex_id: Mandatory** (Every pokedex association must reference a pokedex entry).
card ↔ card_set Each card belongs to a card_set.	Many-to-One Many cards can belong to the same card_set (e.g., multiple cards in the "Base Set"). Each card must belong to exactly one card_set.	**card.set_id: Mandatory** (Every card must belong to a set). **card_set.set_id: Optional** (Not all sets may have cards initially).
card ↔ rarity Each card has a rarity that indicates its scarcity.	Many-to-One Many cards can share the same rarity (e.g., multiple cards can be "Common"). Each card can reference only one rarity.	**card.rarity_id: Optional** (Not all cards have defined rarity, e.g., promo cards). **rarity.rarity_id: Optional** (A rarity may exist without any cards associated initially).
card ↔ supertype Each card has a supertype, which categorizes it broadly (e.g., Pokémon, Trainer, Energy).	Many-to-One Many cards can share the same supertype. Each card must have exactly one supertype.	**card.supertype_id: Mandatory** (Every card must have a supertype). **supertype.supertype_id: Optional** (A supertype can exist even if no cards belong to it yet).
card ↔ type (via card_type) A card can have one or more types (e.g., Grass, Fire).	Many-to-Many A card can have multiple types (e.g., dual-type Pokémon). A type can apply to multiple cards (e.g., multiple Fire-type Pokémon).	**card_type.card_id: Mandatory** (Every type association must belong to a valid card). **card_type.type_id: Mandatory** (Every type association must reference a valid type).

subtype ↔ card (via card_subtype) A card can have one or more subtypes (e.g., "EX", "VSTAR").	Many-to-Many A card can have multiple subtypes. A subtype can apply to multiple cards.	**card_subtype.card_id: Mandatory** (Each subtype must belong to a card). **card_subtype.subtype_id: Mandatory** (Each subtype must belong to a valid subtype).
card_set ↔ legality Each card_set may be legal in specific play formats.	One-to-One Each card_set has one legality record. Each legality record is tied to only one card_set.	**card_set.legality_id: Optional** (Not all sets have legal formats). **legality.legality_id: Optional** (A legality record may exist without referencing a card set).
card ↔ legality (via card_set) Each card can also have legality status (e.g., banned in certain formats).	Many-to-One Many cards can reference the same legality record. Each card can reference only one legality record.	**card.legality_id: Optional** (Not all cards have legality defined). **legality.legality_id: Optional** (Legality may exist without being tied to any card).
price_tracker ↔ card_set (via card) A price_tracker entry references cards that belong to a specific set.	Many-to-One Many price_tracker records can reference the same card_set. Each price_tracker record can only reference one card_set.	**price_tracker.card_id** (and indirectly, set_id): **Mandatory** (Each pricing entry must belong to a set). **card_set.set_id: Optional** (Not all sets may yet have cards in the TCG).

Business Rules for the Pokémon TCG Database

The database is designed to manage information about Pokémon TCG entities for efficient handling of card details, sets, types, and related information. The following are the major **business rules**:

<u>Cards and Their Attributes</u>

- **Card Uniqueness:**
  - One and only one card can exist in the database with the same card_id.
  - Since every card belongs to only one set, for any card, the card_id and set_id are unique.
- **Mandatory Attributes:**
  - All cards must have a name, a set_id, and a supertype.
  - All cards of Pokémon must be associated with a pokedex_card.
- **Optional Attributes:**
  - The cards can have a rarity, artist, flavor_text, and legality.
  - One card can have multiple subtypes or types.

Sets and Legalities

- **Set-Card Relationship:**
  - One card is in exactly one set (set_id); one set may contain multiple cards.
- **Legality Rules:**
  - Every set might provide its legality, including formats it is legal in, such as Standard or Expanded.
  - Every card can have a legality record; this would override any set-based record.
- **Validation Rule:**
  - A set can not be marked as legal for "Standard" if it does not have any cards which match that format.

Rarity and Subtypes

- **Rarity Management:**
  - The card can take one and only one rarity, eg. "Common" or "Rare".
  - Rarities shall be predefined and unique in the rarity table.
- **Subtype Flexibility:**
  - Cards may have multiple subtypes (eg. "EX", "VSTAR").
  - Subtypes should match the supertype of the card:
    - Pokémon card examples would include "EX" or "VSTAR".
    - The subtypes can be "Supporter" and "Stadium" for trainer cards.
- **Validation Rule:**
  - Subtype(s) of the card is as per the permitted subtype(s) of its type.

Pokémon and Types

- **Pokédex Integration:**
  - The cards of the Pokémon have to link to a valid pokedex_card.
  - Cards for Trainers or Energies should not have a reference to the pokedex_card.
- **Card Types (Elements):**
  - A card may have one or more types, including but not limited to Fire and Water.
  - Types shall be previously defined in the type table and linked to cards using card_type.
- **Validation Rule:**
  - A card may not have conflicting types-fire and water, for example-not foresaw by the rules.

Pricing and TCGPlayer Integration

- **Pricing Association:**
  - One price entry per card in the price_tracker table.
  - Pricing needs to include low, mid, high, and market values.
- **Data Update Rule:**
  - Pricing data must be updated periodically; the updated_at field must show the time of last modification.

Relationships and Constraints

- **Supertype and Subtype Exclusivity:**
  - Each card must have exactly one supertype.
  - Subtypes must be consistent with the card's supertype.
- **Mandatory Relationships:**
  - Every card must belong to a set.
  - Cards must reference a supertype, and Pokémon cards must reference the Pokédex.
- **Deletion Cascade:**
  - Deleting a card should automatically remove dependent records (e.g., pricing, types).

Scalability and Data Integrity

- **Adding New Data:**
  - New sets, types, subtypes, or rarities can be added without affecting already existing cards.
  - Additional promotional or exclusive cards can be included with unique identifiers.
- **Referential Integrity:**
  - Relationships between entities should have referential integrity, meaning there should not be any card_type entry without valid card_id and type_id.

End-User and Gameplay Rules

- **User Queries:**
  - This system allows searching cards by set, rarity, legality, or pricing.
- **Format Compliance:**
  - Cards and sets must comply with the up-to-date format rules of Pokémon TCG, for example, legality in Standard, Expanded.
- **Visual Resources:**
  - Each card should have at least one associated image URL (image_url).
  - Large images are preferred to be displayed. Small images are required.

Weak Entities and Multivalued Attributes in the Pokémon TCG Database

Weak Entities

card_type

- **Definition:** Maps the many-to-many relationship between card and subtype, allowing a card to have multiple subtypes (e.g., "EX", "VSTAR").
- **Justification:**
    - Composite Primary Key: Uses card_id (from card) and subtype_id (from subtype) as its primary key.
    - Existence Dependency: Entries in card_subtype cannot exist without a valid card or subtype.
    - Weakness: This table has no meaning independent of its parent entities.

pokedex_card

- **Definition:** Links a card to one or more regional Pokédex entries (e.g., Kanto, Johto).
- **Justification:**
    - Composite Primary Key: Uses card_id (from card) and pokedex_id (from pokedex).
    - Existence Dependency: If a card or Pokédex entry is deleted, all associated pokedex_card records become invalid.

Multivalued Attributes

A **multivalued attribute** may allow an entity for multiple values of a certain property. Relational databases handle these through the process of normalization, though. The following is the list of some multivalued attributes in this database along with their normalized design:

Card types

- **Original Attribute:** A card can have multiple elemental types (e.g., Fire, Psychic).
- **Normalization:**
    - card_type Table: Implements a many-to-many relationship between card and type.
- **Justification:**
    - Eliminates redundancy (no comma-separated types in the card table).
    - Scalable for new types (e.g., "Darkness" or "Fairy").

Card subtypes

- **Original Attribute:** A card may have multiple subtypes (e.g., "EX", "VSTAR").
- **Normalization:**
  - card_subtype Table: Links card to subtype entries.
- **Justification:**
  - Avoids duplicating subtype strings across cards.
  - Enforces referential integrity (all subtypes are predefined in subtype).

Pokedex regions

- **Original Attribute:** A card can appear in multiple regional Pokédexes.
- **Normalization:**
  - pokedex_card Table: Maps cards to their associated Pokédex regions.
- **Justification:**
  - Prevents storing lists of regions in the card table.
  - Allows efficient querying (e.g., "Find all cards in the Galar Pokédex").

Why This Design is Appropriate

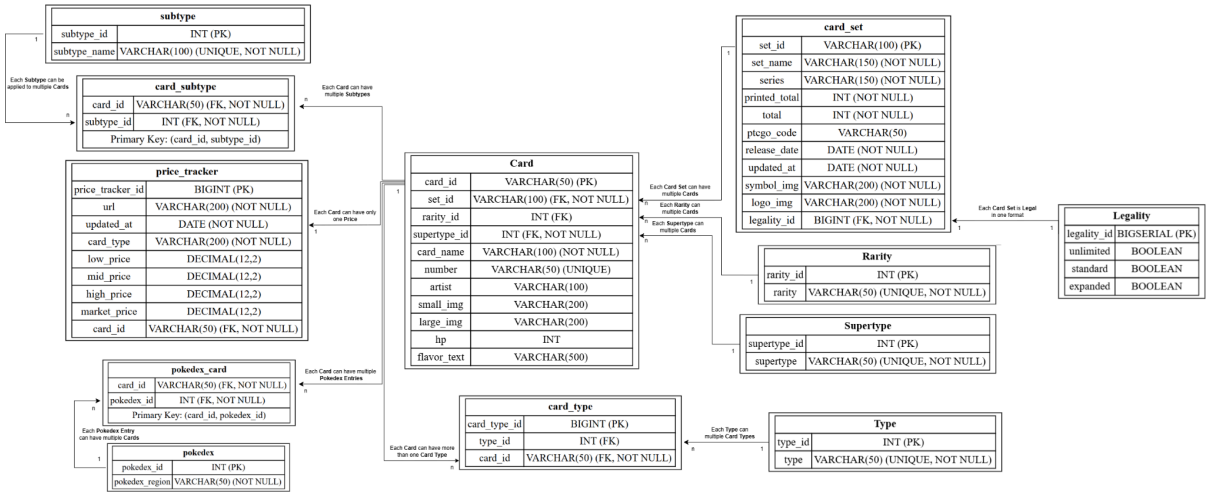
Redundancy Avoidance: Multivalued attributes (types, subtypes) are stored in normalized junction tables, eliminating data duplication.

Scalability: Weak entities like card_subtype and pokedex_card allow seamless expansion (e.g., adding new subtypes or regions).

Data Integrity: Foreign key constraints enforce that all entries in junction tables reference valid parent records.

Query Efficiency: Simple joins retrieve all types/subtypes for a card (e.g., SELECT type FROM type JOIN card_type ON ... WHERE card_id = 'xy1-1').

ER Diagram:



Link for better resolution:
https://drive.google.com/file/d/14sIeeGjIzwe-hFmGqXNhffejebhGD1HO/view?usp=sharing

Business Rules for the Pokémon TCG Database

The database is designed to manage information about Pokémon TCG entities for efficient handling of card details, sets, types, and related information. The following are the major **business rules**:

Cards and Their Attributes

- **Card Uniqueness:**
  - One and only one card can exist in the database with the same card_id.
  - Since every card belongs to only one set, for any card, the card_id and set_id are unique.
- **Mandatory Attributes:**
  - All cards must have a name, a set_id, and a supertype.
  - All cards of Pokémon must be associated with a pokedex_card.
- **Optional Attributes:**
  - The cards can have a rarity, artist, flavor_text, and legality.
  - One card can have multiple subtypes or types.

Sets and Legalities

- **Set-Card Relationship:**
  - One card is in exactly one set (set_id); one set may contain multiple cards.
- **Legality Rules:**
  - Every set might provide its legality, including formats it is legal in, such as Standard or Expanded.
  - Every card can have a legality record; this would override any set-based record.
- **Validation Rule:**
  - A set can not be marked as legal for "Standard" if it does not have any cards which match that format.

Rarity and Subtypes

- **Rarity Management:**
  - The card can take one and only one rarity, eg. "Common" or "Rare".
  - Rarities shall be predefined and unique in the rarity table.
- **Subtype Flexibility:**
  - Cards may have multiple subtypes (eg. "EX", "VSTAR").
  - Subtypes should match the supertype of the card:
    - Pokémon card examples would include "EX" or "VSTAR".
    - The subtypes can be "Supporter" and "Stadium" for trainer cards.
- **Validation Rule:**
  - Subtype(s) of the card is as per the permitted subtype(s) of its type.

Pokémon and Types

- **Pokédex Integration:**
  - The cards of the Pokémon have to link to a valid pokedex_card.
  - Cards for Trainers or Energies should not have a reference to the pokedex_card.
- **Card Types (Elements):**
  - A card may have one or more types, including but not limited to Fire and Water.
  - Types shall be previously defined in the type table and linked to cards using card_type.
- **Validation Rule:**
  - A card may not have conflicting types-fire and water, for example-not foresaw by the rules.

Pricing and TCGPlayer Integration

- **Pricing Association:**
  - One price entry per card in the price_tracker table.
  - Pricing needs to include low, mid, high, and market values.
- **Data Update Rule:**
  - Pricing data must be updated periodically; the updated_at field must show the time of last modification.

Relationships and Constraints

- **Supertype and Subtype Exclusivity:**
  - Each card must have exactly one supertype.
  - Subtypes must be consistent with the card's supertype.
- **Mandatory Relationships:**
  - Every card must belong to a set.
  - Cards must reference a supertype, and Pokémon cards must reference the Pokédex.
- **Deletion Cascade:**
  - Deleting a card should automatically remove dependent records (e.g., pricing, types).

Scalability and Data Integrity

- **Adding New Data:**
  - New sets, types, subtypes, or rarities can be added without affecting already existing cards.
  - Additional promotional or exclusive cards can be included with unique identifiers.
- **Referential Integrity:**
  - Relationships between entities should have referential integrity, meaning there should not be any card_type entry without valid card_id and type_id. This also applies to pokedex_card and card_subtype

End-User and Gameplay Rules

- **User Queries:**
  - This system allows searching cards by set, rarity, legality, or pricing.
- **Format Compliance:**
  - Cards and sets must comply with the up-to-date format rules of Pokémon TCG, for example, legality in Standard, Expanded.
- **Visual Resources:**
  - Each card should have at least one associated image URL (image_url).
  - Large images are preferred to be displayed. Small images are required.

```
CREATE DATABASE IF NOT EXISTS PokeBase;

USE PokeBase;

CREATE TABLE legality (
  legality_id BIGINT AUTO_INCREMENT PRIMARY KEY,
  unlimited BOOLEAN,
  standard BOOLEAN,
  expanded BOOLEAN
);

CREATE TABLE supertype (
  supertype_id INT PRIMARY KEY,
  supertype VARCHAR(50) NOT NULL
);

CREATE TABLE rarity (
  rarity_id INT PRIMARY KEY,
  rarity VARCHAR(50) NOT NULL
);

CREATE TABLE type (
  type_id INT PRIMARY KEY,
  type VARCHAR(50) NOT NULL
);

CREATE TABLE card_set (
  set_id VARCHAR(100) PRIMARY KEY,
  set_name VARCHAR(150) NOT NULL,
  series VARCHAR(150) NOT NULL,
  printed_total INT NOT NULL,
  total INT NOT NULL,
  ptcgo_code VARCHAR(50),
  release_date DATE NOT NULL,
  updated_at DATE NOT NULL,
  symbol_img VARCHAR(200) NOT NULL,
  logo_img VARCHAR(200) NOT NULL,
  legality_id BIGINT NOT NULL,
  CONSTRAINT fk_legality
    FOREIGN KEY(legality_id)
      REFERENCES legality(legality_id)
);
```

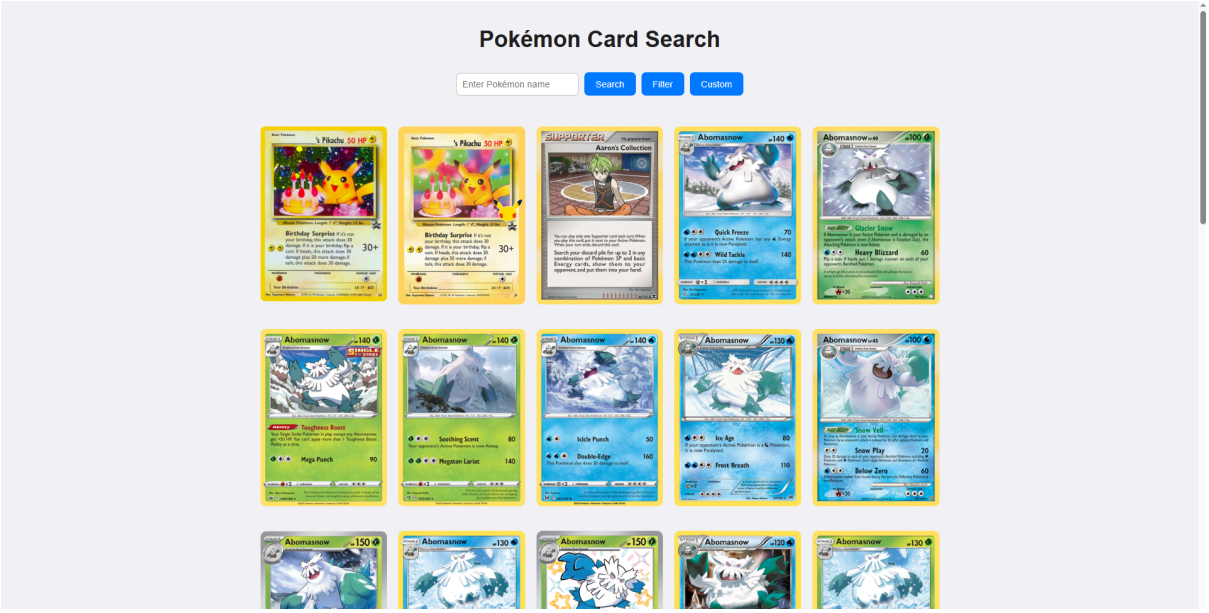
```
CREATE TABLE card (
  card_id VARCHAR(50) PRIMARY KEY,
  card_name VARCHAR(100) NOT NULL,
  number VARCHAR(50) NOT NULL,
  artist VARCHAR(100),
  small_img VARCHAR(200) NOT NULL,
  large_img VARCHAR(200) NOT NULL,
  supertype_id INT NOT NULL,
  set_id VARCHAR(100) NOT NULL,
  rarity_id INT,
  hp INT,
  flavor_text VARCHAR(500),
  CONSTRAINT fk_supertype
    FOREIGN KEY(supertype_id)
      REFERENCES supertype(supertype_id),
  CONSTRAINT fk_set
    FOREIGN KEY(set_id)
      REFERENCES card_set(set_id),
  CONSTRAINT fk_rarity
    FOREIGN KEY(rarity_id)
      REFERENCES rarity(rarity_id)
);

CREATE TABLE pokedex (
  pokedex_id INT PRIMARY KEY,
  pokedex_region VARCHAR(50) NOT NULL
);

CREATE TABLE pokedex_card (
  card_id VARCHAR(50) NOT NULL REFERENCES card(card_id),
  pokedex_id INT NOT NULL REFERENCES pokedex(pokedex_id),
  PRIMARY KEY (card_id, pokedex_id)
);

CREATE TABLE price_tracker (
  price_tracker_id BIGINT AUTO_INCREMENT PRIMARY KEY,
  url VARCHAR(200) NOT NULL,
  updated_at DATE NOT NULL,
  card_type VARCHAR(200) NOT NULL,
  low_price DECIMAL(12, 2),
  mid_price DECIMAL(12, 2),
  high_price DECIMAL(12, 2),
  market_price DECIMAL(12, 2),
  card_id VARCHAR(50) NOT NULL,
  CONSTRAINT fk_card
    FOREIGN KEY(card_id)
      REFERENCES card(card_id)
);
```

```
CREATE TABLE card_type (
  card_type_id BIGINT AUTO_INCREMENT PRIMARY KEY,
  type_id INT,
  card_id VARCHAR(50) NOT NULL,
  CONSTRAINT fk_type_id
    FOREIGN KEY(type_id)
      REFERENCES `type`(type_id),
  CONSTRAINT fk_card_id
    FOREIGN KEY(card_id)
      REFERENCES card(card_id)
);

CREATE TABLE subtype (
  subtype_id INT AUTO_INCREMENT PRIMARY KEY,
  subtype_name VARCHAR(100) UNIQUE NOT NULL
);

CREATE TABLE card_subtype (
  card_id VARCHAR(50) NOT NULL,
  subtype_id INT NOT NULL,
  PRIMARY KEY (card_id, subtype_id),
  FOREIGN KEY (card_id) REFERENCES card(card_id),
  FOREIGN KEY (subtype_id) REFERENCES subtype(subtype_id)
);
```

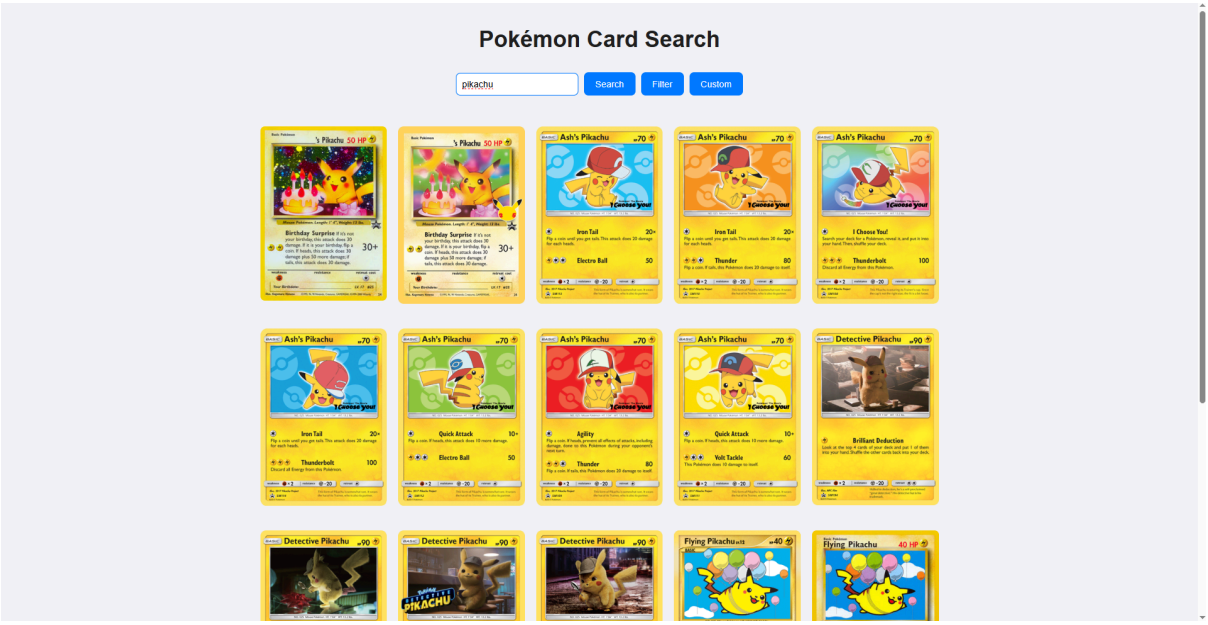
Refer to the Github for the code and installation instructions:
https://github.com/Great64/pokebase

Demonstration of application

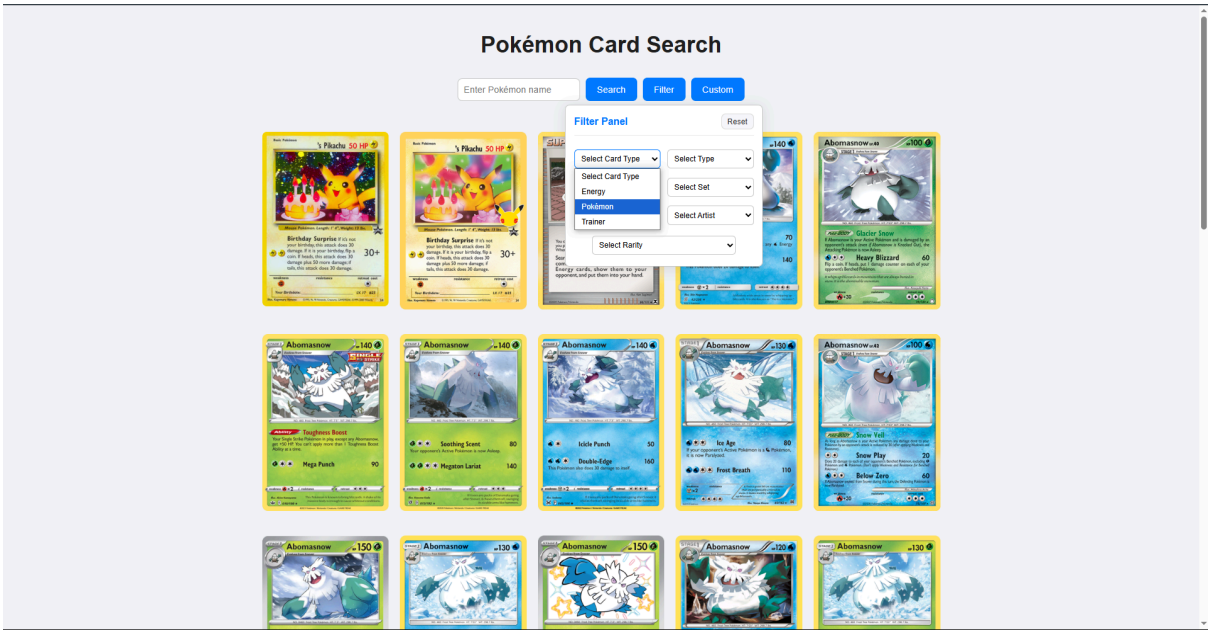
Homepage:

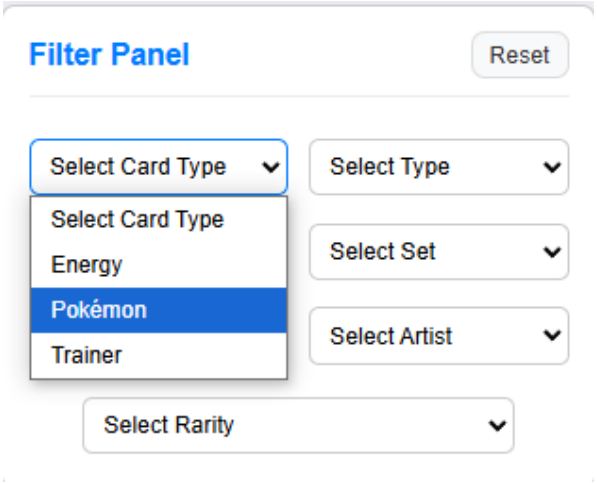


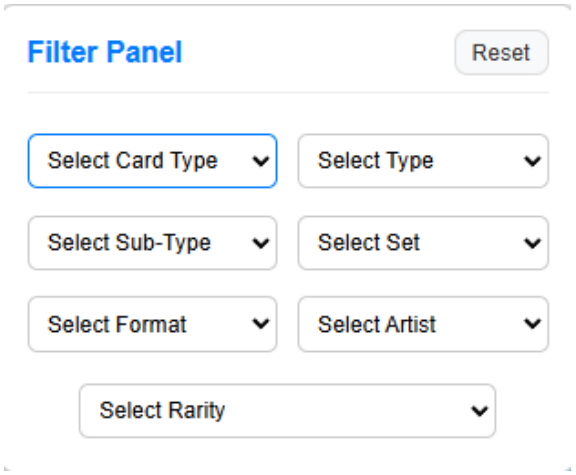
Search:



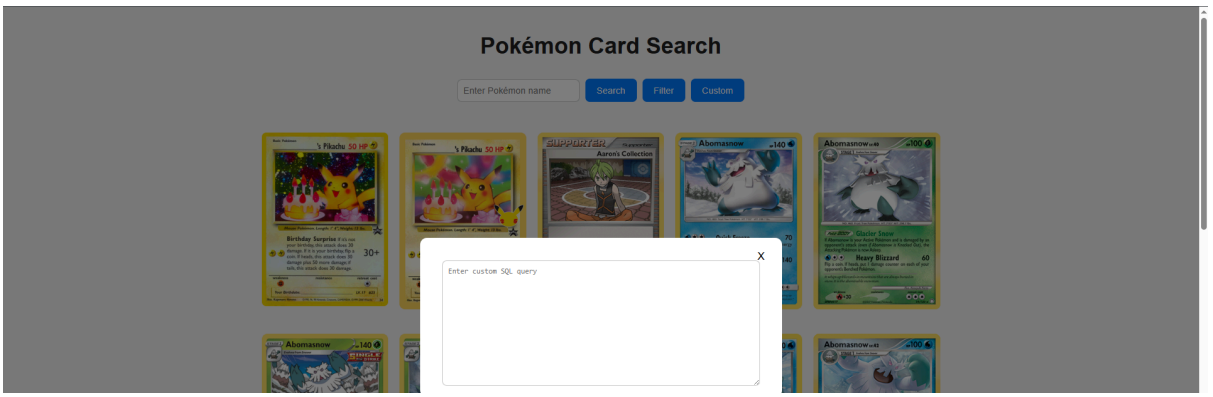
Filters:







Custom Query Compatibility:



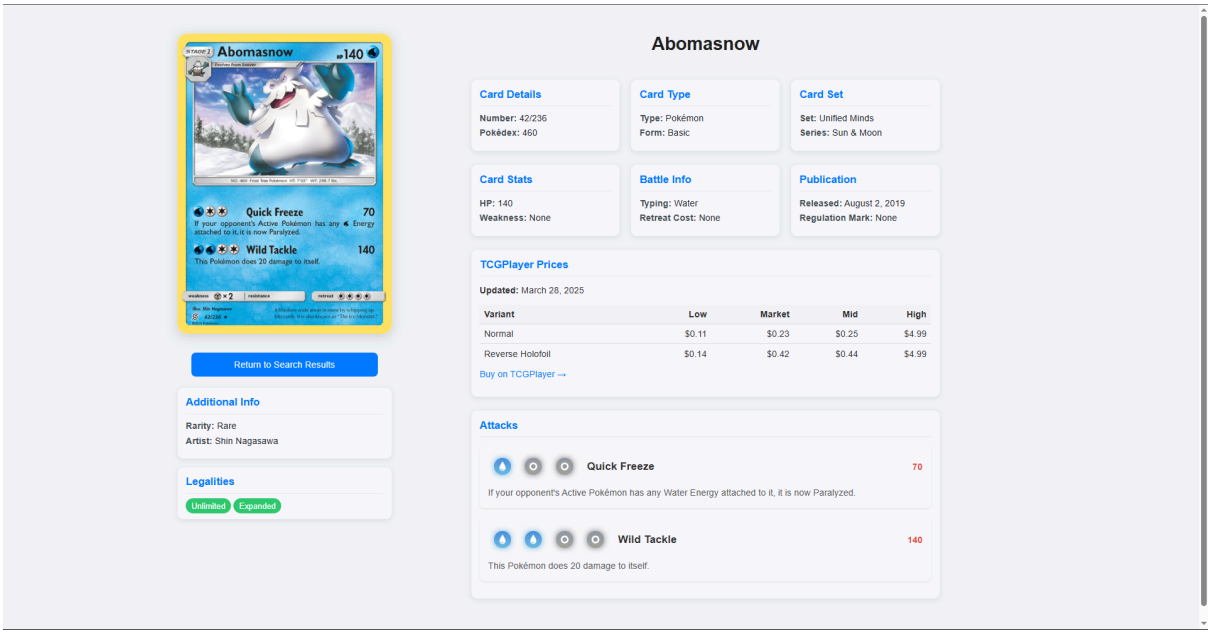
X

Enter custom SQL query

X

```
SELECT card_id, card_name, artist
FROM card
WHERE LOWER(card_name) LIKE '%pikachu%';
```

Card Details:



Return to Search Results

Additional Info

Rarity: Rare

Artist: Shin Nagasawa

Legalities

Unlimited Expanded

Abomasnow

Card Details

Number: 42/236
Pokédex: 460

Card Type

Type: Pokémon
Form: Basic

Card Set

Set: Unified Minds
Series: Sun & Moon

Card Stats

HP: 140
Weakness: None

Battle Info

Typing: Water
Retreat Cost: None

Publication

Released: August 2, 2019
Regulation Mark: None

TCGPlayer Prices

Updated: March 28, 2025

Variant	Low	Market	Mid	High
Normal	$0.11	$0.23	$0.25	$4.99
Reverse Holofoil	$0.14	$0.42	$0.44	$4.99

Buy on TCGPlayer →

Attacks

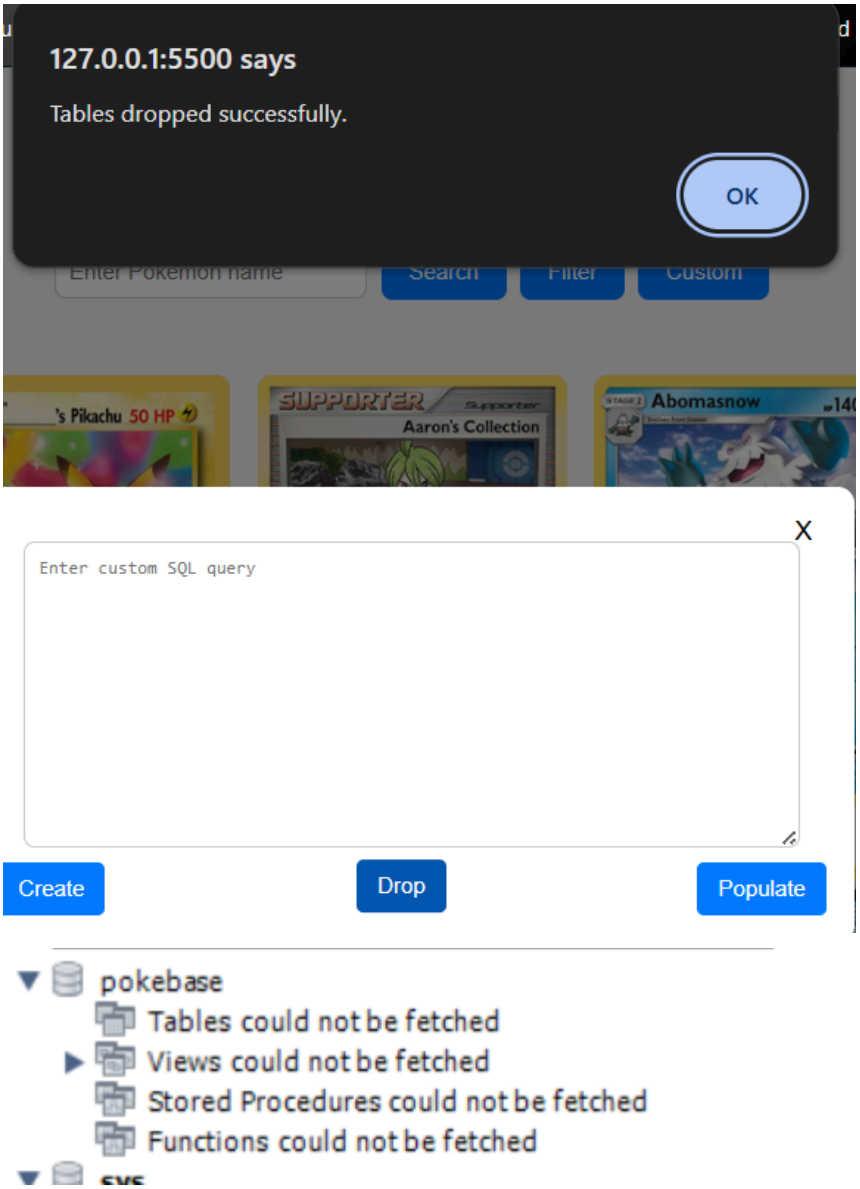
Quick Freeze 70

If your opponent's Active Pokémon has any Water Energy attached to it, it is now Paralyzed.

Wild Tackle 140

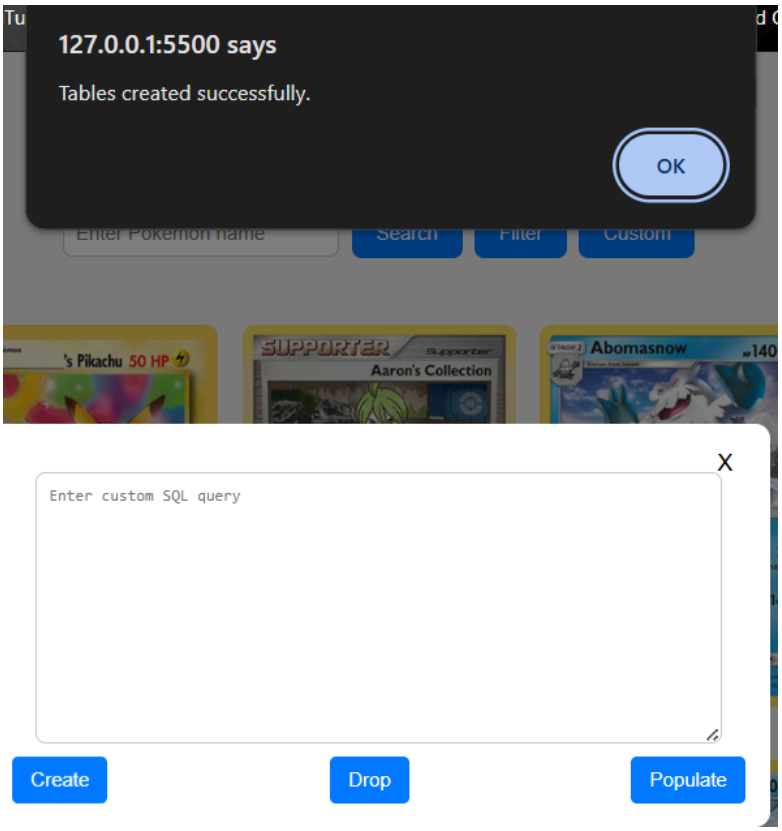
This Pokémon does 20 damage to itself.

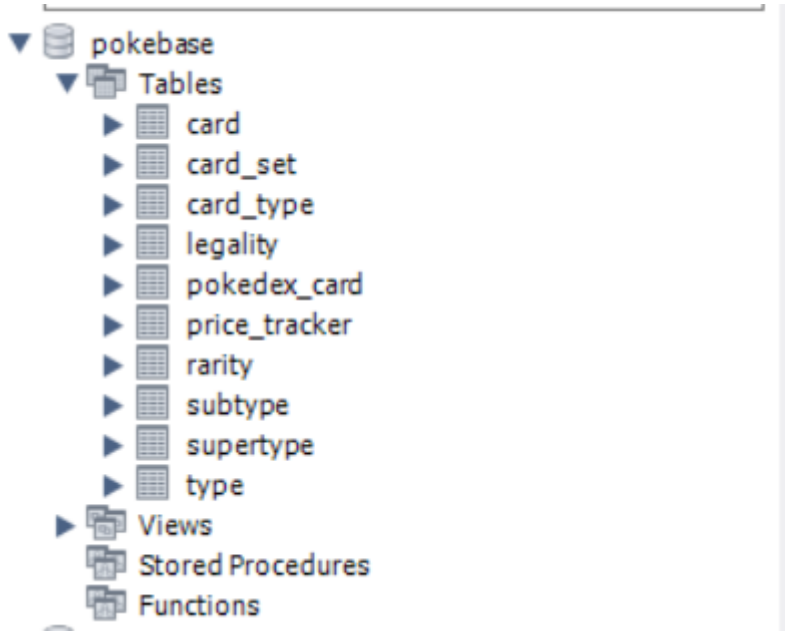
Drop:


127.0.0.1:5500 says
Tables dropped successfully.
OK
Search
Filter
Custom
SUPPORTER
Aaron's Collection
Abomasnow
X
Enter custom SQL query
Create
Drop
Populate

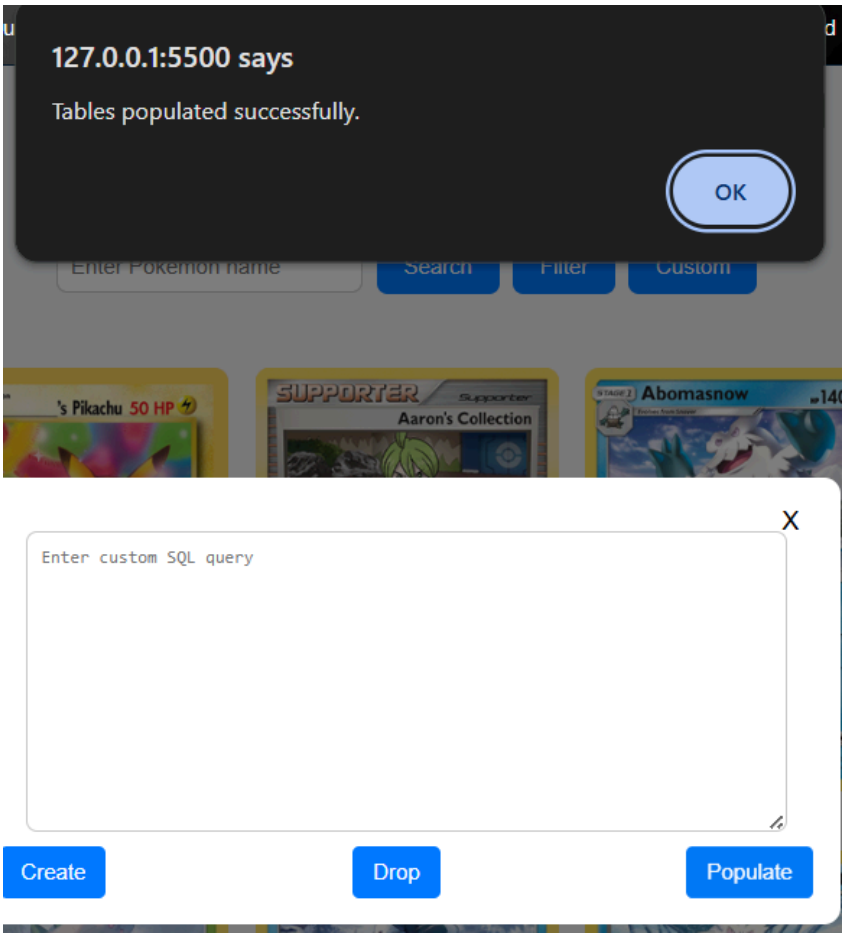
pokebase
Tables could not be fetched
Views could not be fetched
Stored Procedures could not be fetched
Functions could not be fetched

Create:


127.0.0.1:5500 says
Tables created successfully.
OK
Enter Pokemon name
Search
Filter
Custom
's Pikachu 50 HP
SUPPORTER
Aaron's Collection
Abomasnow
X
Enter custom SQL query
Create
Drop
Populate


pokebase
Tables
card
card_set
card_type
legality
pokedex_card
price_tracker
rarity
subtype
supertype
type
Views
Stored Procedures
Functions

Populate:


127.0.0.1:5500 says
Tables populated successfully.
OK
Search
Filter
Custom
's Pikachu 50 HP
SUPPORTER
Aaron's Collection
Abomasnow
X
Enter custom SQL query
Create
Drop
Populate

```
1 •  SELECT * FROM pokebase.card;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap C

card_id	card_name	number	artist	small_img	l
base1-1	Alakazam	1	Ken Sugimori	https://images.pokemontcg.io/base1/1.png	h
base1-10	Mewtwo	10	Ken Sugimori	https://images.pokemontcg.io/base1/10.png	h
base1-100	Lightning Energy	100	Keiji Kinebuchi	https://images.pokemontcg.io/base1/100.png	h
base1-101	Psychic Energy	101	Keiji Kinebuchi	https://images.pokemontcg.io/base1/101.png	h
base1-102	Water Energy	102	Keiji Kinebuchi	https://images.pokemontcg.io/base1/102.png	h
base1-11	Nidoking	11	Ken Sugimori	https://images.pokemontcg.io/base1/11.png	h
base1-12	Ninetales	12	Ken Sugimori	https://images.pokemontcg.io/base1/12.png	h
base1-13	Poliwrath	13	Ken Sugimori	https://images.pokemontcg.io/base1/13.png	h
base1-14	Raichu	14	Ken Sugimori	https://images.pokemontcg.io/base1/14.png	h
base1-15	Venusaur	15	Mitsuhiro Arita	https://images.pokemontcg.io/base1/15.png	h
base1-16	Zapdos	16	Ken Sugimori	https://images.pokemontcg.io/base1/16.png	h

Level of Achievement → ---------- Criteria ↓	**Excellent**	**Good**	**Fair**	**Poor**
Relational Algebra statements (2 marks)	Relational Algebra (RA) statements are clearly documented for all queries. Each query has a correct RA translation that produces the same results as the query. **2 marks**	RA statements are documented for most queries, but some are missing or contain minor issues. A few translations may not accurately reflect the intended query. **1.0 mark**	RA statements are missing for most queries, or major issues exist in the translation. **0.7 mark**	No RA statements are provided. **0 mark**
Documentation including All reports (1 mark)	Final design documents for all assignments are included. Project documentation shows the system design, ER diagram, updated tables, project description, UI screenshots, and installation/operation instructions. Minor elements like some UI screenshots or assumptions may be missing. **1 mark**	Some reports are missing for multiple assignments. Key components such as the ER diagram, final tables, or UI screenshots are incomplete or not updated. **0.7 mark**	Reports and documentation are missing for most or all assignments. **0.5 mark**	Documentation is incomplete or entirely missing. **0.00 mark**