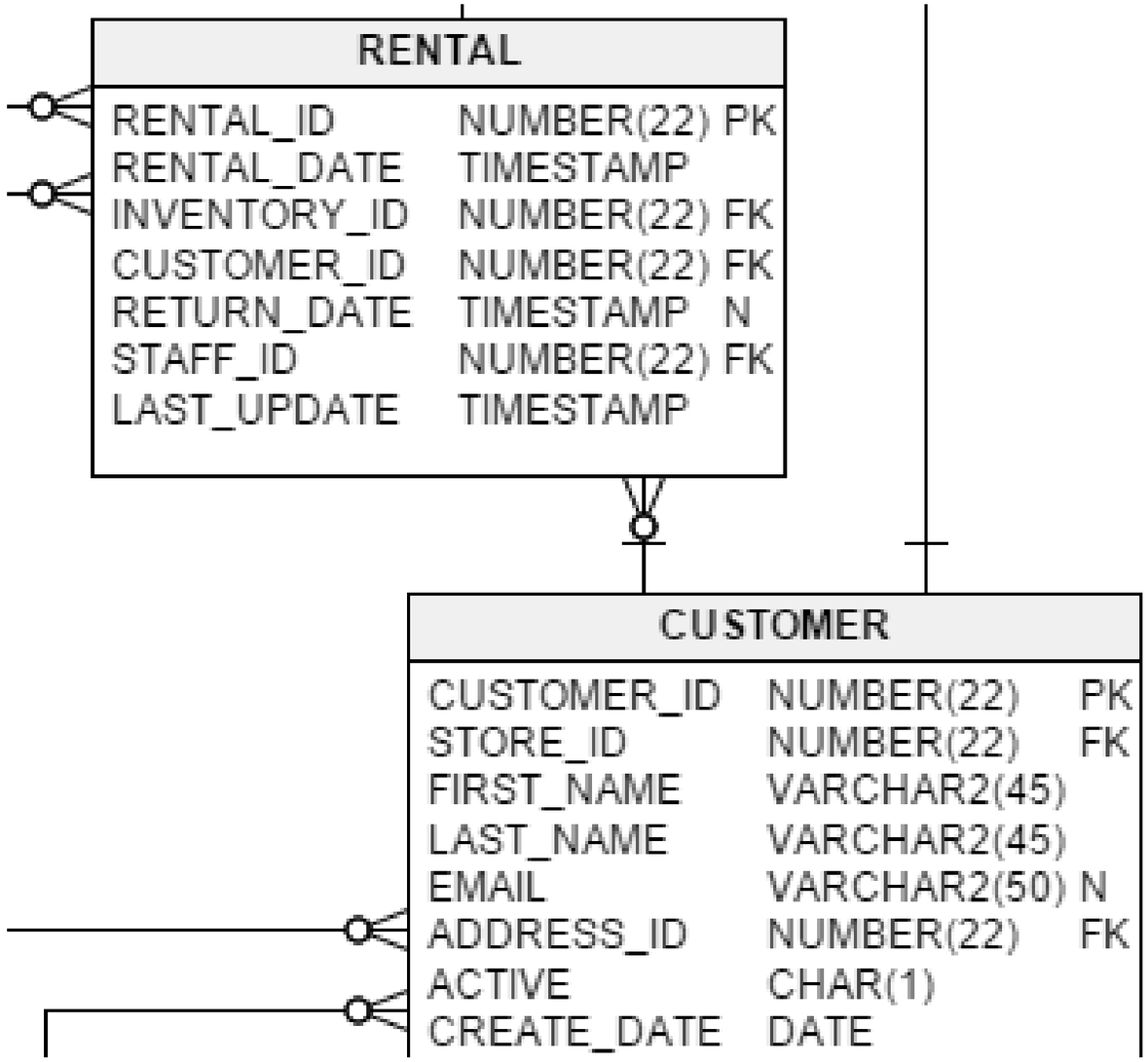
Welcome!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

The Sakila Database



- Highly normalized
- Representative data types
- Custom functions

Topics

- Common data types in PostgreSQL
- Date and time functions and operators
- Parsing and manipulating text
- Full-text search and PostgreSQL Extensions

Common data types

- Text data types
  - `CHAR`, `VARCHAR` and `TEXT`
- Numeric data types
  - `INT` and `DECIMAL`
- Date / time data types
  - `DATE`, `TIME`, `TIMESTAMP`, `INTERVAL`
- Arrays

Text data types

```
SELECT title
FROM film
LIMIT 5
```

```
+------------------+
| title            |
|------------------|
| ACADEMY DINOSAUR |
| ACE GOLDFINGER   |
| ADAPTATION HOLES |
| AFFAIR PREJUDICE |
| AFRICAN EGG      |
+------------------+
```

```
SELECT description
FROM film
LIMIT 2
```

```
+----------------------------------------+
| description                            |
|----------------------------------------|
| A Epic Drama of a Feminist And a Mad   |
|  Scientist who must Battle a Teacher in |
|  The Canadian Rockies.                 |
| A Astounding Epistle of a Database     |
|  Administrator And a Explorer who      |
|  must Find a Car in Ancient China      |
+----------------------------------------+
```

Numeric data types

```
SELECT
    payment_id
FROM payment
LIMIT 5
```

```
+-------------+
| payment_id  |
|-------------|
| 1           |
| 2           |
| 3           |
| 4           |
| 5           |
+-------------+
```

```
SELECT
    amount
FROM payment
LIMIT 5
```

```
+--------+
| amount |
|--------|
| 2.99   |
| 0.99   |
| 5.99   |
| 0.99   |
| 9.99   |
+--------+
```

Determining data types from existing tables

```
SELECT
    title,
    description,
    special_features
FROM FILM
LIMIT 5
```

```
+---------------+------------------+------------------------------+
| title         |  description     | special_features             |
|---------------|------------------|------------------------------|
| ACADEMY D...  | A Epic...        | {Deleted Scenes,Behi...}     |
| ACE GOLD...   | A Astound..      | {Trailers,Deleted Scenes}    |
| AFFAIR PR...  | A Fanciful,..    | {Commentaries,Behind the...} |
+---------------+------------------+------------------------------+
```

Determining data types from existing tables

```
SELECT
    column_name,
    data_type
FROM INFORMATION_SCHEMA.COLUMNS
WHERE column_name in ('title','description','special_features')
  AND table_name ='film';
```

```
+------------------+-------------------+
| column_name      | data_type         |
|------------------|-------------------|
| title            | character varying |
| description      | text              |
| special_features | ARRAY             |
+------------------+-------------------+
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Date and time data types

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

TIMESTAMP data types

- ISO 8601 format: yyyy-mm-dd

```
+-----------------------------+
| timestamp                   |
|-----------------------------|
| 2019-03-26 01:05:17.93027+00 |
+-----------------------------+
```

```sql
SELECT payment_date
FROM payment;
```

```
+--------------------+
| payment_date       |
|--------------------|
| 2005-05-25 11:30:37 |
+--------------------+
```

DATE and TIME data types

```
+------------+-------------------+
| date       | time              |
|------------|-------------------|
| 2005-05-28 | 01:05:17.93027+00 |
+------------+-------------------+
```

```
SELECT create_date
FROM customer
```

```
+-------------+
| create_date |
|-------------|
| 2006-02-14  |
+-------------+
```

INTERVAL data types

```
+-----------+
| interval  |
|-----------|
| 4 days    |
+-----------+
```

```sql
SELECT rental_date + INTERVAL '3 days' as expected_return
FROM rental;
```

```
+---------------------+
| expected_return     |
|---------------------|
| 2005-05-27 22:53:30 |
+---------------------+
```

Looking at date and time types

```
SELECT
    column_name,
    data_type
FROM INFORMATION_SCHEMA.COLUMNS
WHERE column_name in ('rental_date')
  AND table_name ='rental';
```

```
+-------------+-----------------------------+
| column_name | data_type                   |
|-------------|-----------------------------|
| rental_date | timestamp without time zone |
+-------------+-----------------------------+
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Working with ARRAYs

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

Before we get started

CREATE TABLE example

```
CREATE TABLE my_first_table (
    first_column text,
    second_column integer
);
```

INSERT example

```
INSERT INTO my_first_table
    (first_column, second_column) VALUES ('text value', 12);
```

ARRAY a special type

Let's create a simple table with two array columns.

```
CREATE TABLE grades (
    student_id int,
    email text[][],
    test_scores int[]
);
```

INSERT statements with ARRAYS

Example INSERT statement:

```
INSERT INTO grades
    VALUES (1,
    '{{"work","work1@datacamp.com"},{"other","other1@datacamp.com"}}',
    '{92,85,96,88}' );
```

Accessing ARRAYs

```
SELECT
   email[1][1] AS type,
   email[1][2] AS address,
   test_scores[1],
FROM grades;
```

```
+--------+--------------------+-------------+
| type   |  address           | test_scores |
|--------|--------------------|-------------|
| work   | work1@datacamp.com | 92          |
| work   | work2@datacamp.com | 76          |
+--------+--------------------+-------------+
```

Note that PostgreSQL array indexes start with one and not zero.

Searching ARRAYs

```
SELECT
   email[1][1] as type,
   email[1][2] as address,
   test_scores[1]
FROM grades
WHERE email[1][1] = 'work';
```

```
+--------+--------------------+-------------+
| type   |  address           | test_scores |
|--------|--------------------|-------------|
| work   | work1@datacamp.com | 92          |
| work   | work2@datacamp.com | 76          |
+--------+--------------------+-------------+
```

ARRAY functions and operators

```
SELECT
   email[2][1] as type,
   email[2][2] as address,
   test_scores[1]
FROM grades
WHERE 'other' = ANY (email);
```

```
+---------+---------------------+-------------+
| type    |  address            | test_scores |
|---------|-------------------------------------|
| other   | other1@datacamp.com | 92          |
| null    | null                | 76          |
+---------+---------------------+-------------+
```

ARRAY functions and operators

```
SELECT
   email[2][1] as type,
   email[2][2] as address,
   test_scores[1]
FROM grades
WHERE email @> ARRAY['other'];
```

```
+---------+----------------------+-------------+
| type    |  address             | test_scores |
|---------|-----------------------------------|
| other   | other1@datacamp.com | 92          |
| null    | null                 | 76          |
+---------+----------------------+-------------+
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Overview of basic arithmetic operators

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

Topics

- Overview of basic arithmetic operators
- The `CURRENT_DATE`, `CURRENT_TIMESTAMP`, `NOW()` functions
- The `AGE()` function
- The `EXTRACT()`, `DATE_PART()`, and `DATE_TRUNC()` functions

Adding and subtracting date / time data

```
SELECT date '2005-09-11' - date '2005-09-10';
```

```
+----------+
| integer  |
|----------|
| 1        |
+----------+
```

Adding and subtracting date / time data

```
SELECT date '2005-09-11' + integer '3';
```

```
+------------+
| date       |
|------------|
| 2005-09-14 |
+------------+
```

Adding and subtracting date / time data

```
SELECT date '2005-09-11 00:00:00' - date '2005-09-09 12:00:00';
```

```
+-----------------+
| interval        |
|-----------------|
| 1 day 12:00:00 |
+-----------------+
```

Calculating time periods with AGE

```
SELECT AGE(timestamp '2005-09-11 00:00:00', timestamp '2005-09-09 12:00:00');
```

```
+-----------------+
| interval        |
|-----------------|
| 1 day 12:00:00 |
+-----------------+
```

DVDs, really??

```
SELECT
    AGE(rental_date)
FROM rental;
```

```
+---------------------------------+
| age                             |
|---------------------------------|
| 13 years 11 mons 12 days 01:06:30 |
| 13 years 11 mons 12 days 01:05:27 |
| 13 years 11 mons 12 days 00:56:21 |
+---------------------------------+
```

Date / time arithmetic using INTERVALs

```
SELECT rental_date + INTERVAL '3 days' as expected_return
FROM rental;
```

```
+---------------------+
| expected_return     |
|---------------------|
| 2005-05-27 22:53:30 |
+---------------------+
```

Date / time arithmetic using INTERVALs

```
SELECT timestamp '2019-05-01' + 21 * INTERVAL '1 day';
```

```
+----------------------------+
| timestamp without timezone |
|----------------------------|
| 2019-05-22 00:00:00        |
+----------------------------+
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Functions for retrieving current date/time

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

Retrieving the current timestamp

```
SELECT NOW();
```

```
+-------------------------------+
| now()                         |
|-------------------------------|
| 2019-04-19 02:51:18.448641+00 |
+-------------------------------+
```

Retrieving the current timestamp

```
SELECT NOW()::timestamp;
```

```
+----------------------------+
| now()                      |
|----------------------------|
| 2019-04-19 02:51:18.448641 |
+----------------------------+
```

Retrieving the current timestamp

PostgreSQL specific casting

```
SELECT NOW()::timestamp;
```

CAST() function

```
SELECT CAST(NOW() as timestamp);
```

Retrieving the current timestamp

```
SELECT CURRENT_TIMESTAMP;
```

```
+-------------------------------+
| current_timestamp             |
|-------------------------------|
| 2019-04-19 02:51:18.448641+00 |
+-------------------------------+
```

Retrieving the current timestamp

```
SELECT CURRENT_TIMESTAMP(2);
```

```
+-------------------------------+
| current_timestamp             |
|-------------------------------|
| 2019-04-19 02:51:18.44+00     |
+-------------------------------+
```

Current date and time

```
SELECT CURRENT_DATE;
```

```
+--------------+
| current_date |
|--------------|
| 2019-04-19   |
+--------------+
```

Current date and time

```
SELECT CURRENT_TIME;
```

```
+-------------------------+
| current_time            |
|-------------------------|
| 04:06:30.929845+00:00   |
+-------------------------+
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Extracting and transforming date / time data

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

Extracting and transforming date and time data

Exploring the `EXTRACT()`, `DATE_PART()` and `DATE_TRUNC()` functions

- Transactional timestamp precision not useful for analysis

```
2005-05-13 08:53:53
```

- Often need to extract parts of timestamps

```
2005 or 5 or 2 or Friday
```

- Or convert / truncate timestamp precision to standardize

```
2005-05-13 00:00:00
```

Extracting and transforming date / time data

- EXTRACT(**field** *FROM* **source**)

```
SELECT EXTRACT(quarter FROM timestamp '2005-01-24 05:12:00') AS quarter;
```

- DATE_PART(**'field'**, **source**)

```
SELECT DATE_PART('quarter', timestamp '2005-01-24 05:12:00') AS quarter;
```

```
+---------+
| quarter |
|---------|
| 1       |
+---------+
```

Extracting sub-fields from timestamp data

Transactional data from DVD Rentals *payment* table

```
SELECT * FROM payment;
```

```
+------------------------------------------------------------------------------+
| payment_id | customer_id | staff_id | rental_id | amount | payment_date         |
|------------|-------------|----------|-----------|--------|----------------------|
| 1          | 1           | 1        | 76        | 2.99   | 2005-05-25 11:30:37 |
| 2          | 1           | 1        | 573       | 0.99   | 2005-05-28 10:35:23 |
| 3          | 1           | 1        | 1185      | 5.99   | 2005-06-15 0:54:12  |
+------------------------------------------------------------------------------+
```

Extracting sub-fields from timestamp data

Data from *payment* table by year and quarter

```
SELECT
  EXTRACT(quarter FROM payment_date) AS quarter,
  EXTRACT(year FROM payment_date) AS year,
  SUM(amount) AS total_payments
FROM
  payment
GROUP BY 1, 2;
```

Results

```
+---------------------------------+
| quarter | year | total_payments |
|---------|------|----------------|
| 2       | 2005 | 14456.31       |
| 3       | 2005 | 52446.02       |
| 1       | 2006 | 514.18         |
+---------------------------------+
```

Truncating timestamps using DATE_TRUNC()

The `DATE_TRUNC()` function will truncate timestamp or interval data types.

- Truncate timestamp '2005-05-21 15:30:30' by year

```
SELECT DATE_TRUNC('year', TIMESTAMP '2005-05-21 15:30:30');
```

```
Result: 2005-01-01 00:00:00
```

- Truncate timestamp '2005-05-21 15:30:30' by month

```
SELECT DATE_TRUNC('month', TIMESTAMP '2005-05-21 15:30:30');
```

```
Result: 2005-05-01 00:00:00
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Reformatting string and character data

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

Topics

- Reformatting string and character data.
- Parsing string and character data.
- Determine string length and character position.
- Truncating and padding string data.

The string concatenation operator

```
SELECT
  first_name,
  last_name,
  first_name || ' ' || last_name AS full_name
FROM customer
```

```
+------------+-----------+--------------------+
| first_name | last_name | full_name          |
|------------|-----------|--------------------|
| MARY       | SMITH     | MARY SMITH         |
| LINDA      | WILLIAMS  | LINDA WILLIAMS     |
+------------+-----------+--------------------+
```

String concatenation with functions

```
SELECT
  CONCAT(first_name,' ', last_name) AS full_name
FROM customer;
```

```
+-----------------------------------------+
| first_name | last_name | full_name      |
|-----------------------------------------|
| MARY       | SMITH     | MARY SMITH     |
| LINDA      | WILLIAMS  | LINDA WILLIAMS |
+-----------------------------------------+
```

String concatenation with a non-string input

```
SELECT
  customer_id || ': '
  || first_name || ' '
  || last_name AS full_name
FROM customer;
```

```
+--------------------+
| full_name          |
|--------------------|
| 1: MARY SMITH      |
| 2: LINDA WILLIAMS  |
+--------------------+
```

Changing the case of string

```
SELECT
  UPPER(email)
FROM customer;
```

```
+-----------------------------------+
| UPPER(email)                      |
|-----------------------------------|
| MARY.SMITH@SAKILACUSTOMER.ORG     |
| PATRICIA.JOHNSON@SAKILACUSTOMER.ORG |
| LINDA.WILLIAMS@SAKILACUSTOMER.ORG |
+-----------------------------------+
```

Changing the case of string

```
SELECT
  LOWER(title)
FROM film;
```

```
+--------------------+
| LOWER(title)       |
|--------------------|
| academy dinosaur   |
| ace goldfinger     |
| adaptation holes   |
+--------------------+
```

Changing the case of string

```
SELECT
  INITCAP(title)
FROM film;
```

```
+--------------------+
| INITCAP(title)     |
|--------------------|
| Academy Dinosaur   |
| Ace Goldfinger     |
| Adaptation Holes   |
+--------------------+
```

Replacing characters in a string

```
SELECT description FROM film;
```

```
+-------------------------------------------------+
| description                                     |
|-------------------------------------------------|
| A Epic Drama of a Feminist And a Mad Scientist...       |
| A Astounding Epistle of a Database Administrator...     |
| A Astounding Reflection of a Lumberjack And a Car...    |
| A Fanciful Documentary of a Frisbee And a Lumberjack... |
| A Fast-Paced Documentary of a Pastry Chef And a...       |
+-------------------------------------------------+
```

Replacing characters in a string

```
SELECT
    REPLACE(description, 'A Astounding',
                'An Astounding') as description
FROM film;
```

```
+-----------------------------------------------------+
| description                                         |
|-----------------------------------------------------|
| A Epic Drama of a Feminist And a Mad Scientist...   |
| An Astounding Epistle of a Database Administrator...|
| An Astounding Reflection of a Lumberjack And a Car...|
+-----------------------------------------------------+
```

Manipulating string data with REVERSE

```
SELECT
  title,
  REVERSE(title)
FROM
  film AS f;
```

```
+-----------------------------------+
| title              | reverse(title)   |
|-----------------------------------|
| ACADEMY DINOSAUR | RUASONID YMEDACA |
| ACE GOLDFINGER   | REGNIFDLOG ECA   |
+-----------------------------------+
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Parsing string and character data

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

Determining the length of a string

```
SELECT
    title,
    CHAR_LENGTH(title)
FROM film;
```

```
+------------------+----------------------+
| title            | CHAR_LENGTH(title)   |
|------------------+----------------------|
| ACADEMY DINOSAUR | 16                   |
| ACE GOLDFINGER   | 14                   |
| ADAPTATION HOLES | 16                   |
+------------------+----------------------+
```

Determining the length of a string

```
SELECT
    title,
    LENGTH(title)
FROM film;
```

```
+--------------------+-----------------+
| title              | LENGTH(title)   |
|--------------------+-----------------|
| ACADEMY DINOSAUR   | 16              |
| ACE GOLDFINGER     | 14              |
| ADAPTATION HOLES   | 16              |
+--------------------+-----------------+
```

Finding the position of a character in a string

```
SELECT
    email,
    POSITION('@' IN email)
FROM customer;
```

```
+-----------------------------------+-------------------------+
| email                             | POSITION('@' IN email) |
|-----------------------------------|-------------------------|
| MARY.SMITH@sakilacustomer.org     | 11                      |
| PATRICIA.JOHNSON@sakilacustomer.org | 17                    |
| LINDA.WILLIAMS@sakilacustomer.org | 15                      |
+-----------------------------------+-------------------------+
```

Finding the position of a character in a string

```
SELECT
    email,
    STRPOS(email, '@')
FROM customer;
```

```
+-----------------------------------+--------------------+
| email                             | STRPOS(email, '@') |
|-----------------------------------|--------------------|
| MARY.SMITH@sakilacustomer.org     | 11                 |
| PATRICIA.JOHNSON@sakilacustomer.org | 17               |
| LINDA.WILLIAMS@sakilacustomer.org | 15                 |
+-----------------------------------+--------------------+
```

Parsing string data

```
SELECT
    LEFT(description, 50)
FROM film;
```

```
+--------------------------------------------------+
| description                                      |
|--------------------------------------------------|
| A Epic Drama of a Feminist And a Mad Scientist who |
| A Astounding Epistle of a Database Administrator A |
| A Astounding Reflection of a Lumberjack And a Car  |
+--------------------------------------------------+
```

Parsing string data

```
SELECT
    RIGHT(description, 50)
FROM film;
```

```
+--------------------------------------------------+
| description                                      |
|--------------------------------------------------|
|  who must Battle a Teacher in The Canadian Rockies |
| nd a Explorer who must Find a Car in Ancient China |
| Car who must Sink a Lumberjack in A Baloon Factory |
+--------------------------------------------------+
```

Extracting substrings of character data

```
SELECT
    SUBSTRING(description, 10, 50)
FROM
  film AS f;
```

```
+----------------------------------------------------+
| description                                        |
|----------------------------------------------------|
| ama of a Feminist And a Mad Scientist who must Bat |
| ing Epistle of a Database Administrator And a Expl |
| ing Reflection of a Lumberjack And a Car who must  |
+----------------------------------------------------+
```

Extracting substrings of character data

```
SELECT
    SUBSTRING(email FROM 0 FOR POSITION('@' IN email))
FROM
  customer;
```

```
+-----------------------------------------------+
| SUBSTRING(email FROM 0 FOR POSITION('@' IN email)) |
|-----------------------------------------------|
| MARY.SMITH                                    |
| PATRICIA.JOHNSON                              |
| LINDA.WILLIAMS                                |
+-----------------------------------------------+
```

Extracting substrings of character data

```
SELECT
    SUBSTRING(email FROM POSITION('@' IN email)+1 FOR CHAR_LENGTH(email))
FROM
  customer;
```

```
+-----------------------------------------------------------------+
| SUBSTRING(email FROM POSITION('@' IN email)+1 FOR CHAR_LENGTH(email)) |
|-----------------------------------------------------------------|
| sakilacustomer.org                                              |
| sakilacustomer.org                                              |
| sakilacustomer.org                                              |
+-----------------------------------------------------------------+
```

Extracting substrings of character data

```
SELECT
    SUBSTR(description, 10, 50)
FROM
  film AS f;
```

```
+----------------------------------------------------+
| description                                        |
|----------------------------------------------------|
| ama of a Feminist And a Mad Scientist who must Bat |
| ing Epistle of a Database Administrator And a Expl |
| ing Reflection of a Lumberjack And a Car who must  |
+----------------------------------------------------+
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Truncating and padding string data

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

Removing whitespace from strings

```
TRIM([leading | trailing | both] [characters] from string)
```

- **First parameter:** [leading | trailing | both]
- **Second parameter:** [characters]
- **Third parameter:** from string

Removing whitespace from strings

```
SELECT TRIM('  padded  ');
```

```
+--------+
| TRIM   |
|--------|
| padded |
+--------+
```

Removing whitespace from strings

```
SELECT LTRIM('  padded   ');
```

```
+------------+
| LTRIM      |
|------------|
| padded     |
+------------+
```

Removing whitespace from strings

```
SELECT RTRIM('   padded    ');
```

```
+----------+
| RTRIM    |
|----------|
|   padded |
+----------+
```

Padding strings with character data

```
SELECT LPAD('padded', 10, '#');
```

```
+-------------+
| LPAD        |
|-------------|
| ####padded  |
+-------------+
```

Padding strings with whitespace

```
SELECT LPAD('padded', 10);
```

```
+-------------+
| LPAD        |
|-------------|
|     padded  |
+-------------+
```

```
SELECT LPAD('padded', 5);
```

```
+------------+
| LPAD       |
|------------|
|     padde  |
+------------+
```

Padding strings with whitespace

```
SELECT RPAD('padded', 10, '#');
```

```
+-------------+
| RPAD        |
|-------------|
| padded####  |
+-------------+
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Introduction to full-text search

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

Topics

- Full Text search
- Extending PostgreSQL
- Improving full text search with extensions

The LIKE operator

_ wildcard: Used to match exactly one character.

% wildcard: Used to match zero or more characters.

```
SELECT title
FROM film
WHERE title LIKE 'ELF%';
```

```
+----------------------+
| title                |
+----------------------+
| ELF PARTY            |
+----------------------+
```

The LIKE operator

```
SELECT title
FROM film
WHERE title LIKE '%ELF';
```

```
+----------------------+
| title                |
+----------------------+
| ENCINO ELF           |
| GHOSTBUSTERS ELF     |
+----------------------+
```

The LIKE operator

```
SELECT title
FROM film
WHERE title LIKE '%elf%';
```

```
+----------------------+
| title                |
+----------------------+
```

LIKE versus full-text search

```
SELECT title, description
FROM film
WHERE to_tsvector(title) @@ to_tsquery('elf');
```

```
+----------------------+
| title                |
+----------------------+
| ELF PARTY            |
| ENCINO ELF           |
| GHOSTBUSTERS ELF     |
+----------------------+
```

What is full-text search?

Full text search provides a means for performing natural language queries of text data in your database.

- Stemming
- Spelling mistakes
- Ranking

Full-text search syntax explained

```
SELECT title, description
FROM film
WHERE to_tsvector(title) @@ to_tsquery('elf');
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Extending PostgreSQL

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

User-defined data types

Enumerated data types

```
CREATE TYPE dayofweek AS ENUM (
      'Monday',
      'Tuesday',
      'Wednesday',
      'Thursday',
      'Friday',
      'Saturday',
      'Sunday'
);
```

Getting information about user-defined data types

```
SELECT typname, typcategory
FROM pg_type
WHERE typname='dayofweek';
```

```
+-----------+-------------+
| typname   | typcategory |
|-----------|-------------|
| dayofweek | E           |
+-----------+-------------+
```

Getting information about user-defined data types

```
SELECT column_name, data_type, udt_name
FROM INFORMATION_SCHEMA.COLUMNS
WHERE table_name ='film';
```

```
+-------------------------------------------+
| column_name | data_type         | udt_name    |
|-------------|-------------------|-------------|
| title       | character varying | varchar     |
| rating      | USER-DEFINED      | mpaa_rating |
+-------------------------------------------+
```

User-defined functions

```
CREATE FUNCTION squared(i integer) RETURNS integer AS $$
    BEGIN
        RETURN i * i;
    END;
$$ LANGUAGE plpgsql;
```

```
SELECT squared(10);
```

```
+---------+
| squared |
|---------|
| 100     |
+---------+
```

User-defined functions in the Sakila database

- **get_customer_balance(customer_id, effective_data):** calculates the current outstanding balance for a given customer.
- **inventory_held_by_customer(inventory_id):** returns the customer_id that is currently renting an inventory item or null if it's currently available.
- **inventory_in_stock(inventory_id):** returns a boolean value of whether an inventory item is currently in stock.

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Intro to PostreSQL extensions

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

Intro to PostgreSQL extensions

Commonly used extensions

- PostGIS
- PostPic
- fuzzystrmatch
- pg_trgm

Querying extension meta data

Available Extensions

```
SELECT name
FROM pg_available_extensions;
```

```
+--------------------+
| name               |
|--------------------|
| dblink             |
| pg_stat_statements |
+--------------------+
```

Installed Extensions

```
SELECT extname
FROM pg_extension;
```

```
+----------+
| name     |
|----------|
| plpgsql  |
+----------+
```

```
--Enable the fuzzystrmatch extension
CREATE EXTENSION IF NOT EXISTS fuzzystrmatch;
--Confirm that fuzzstrmatch has been enabled
SELECT extname FROM pg_extension;
```

```
+----------------+
| name           |
|----------------|
| plpgsql        |
| fuzzystrmatch  |
+----------------+
```

Using fuzzystrmatch or fuzzy searching

```
SELECT levenshtein('GUMBO', 'GAMBOL');
```

```
+-------------+
| levenshtein |
|-------------|
|           2 |
+-------------+
```

Compare two strings with pg_trgm

```
SELECT similarity('GUMBO', 'GAMBOL');
```

```
+------------+
| similarity |
|------------|
| 0.18181818 |
+------------+
```

Let's practice!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Putting it All Together

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL

Brian Piccolo
Sr. Director, Digital Strategy

datacamp

Functions for manipulating data recap and review

- Common data types in PostgreSQL
- Date/time functions and operators
- Parsing and manipulating text
- PostgreSQL Extensions and full-text search

Thank you!

FUNCTIONS FOR MANIPULATING DATA IN POSTGRESQL