

# Piedmont Housing Alliance Data Analysis

1/25/23

```
# Load data management and analysis tools

library(tidyverse)
library(tidycensus)
library(readxl)
library(janitor)
library(lubridate)

# Load spatial analysis tools

library(tidygeocoder)
library(sf)
library(tigris)
library(mapview)

# Load data visualization tools

library(hdatools)
library(scales)
library(ggtext)
library(ggridges)
library(kableExtra)
library(reactable)
library(patchwork)
library(formattable)

# Set defaults

setwd("~/repos/piedmont")
```

```
today <- Sys.Date()
```

### Please note

This document is still under construction. It was last updated January 25, 2023 at 01:20 PM.

## About

Piedmont Housing Alliance (PHA) is a 501(c)3 nonprofit organization that provides housing services to clients in the Charlottesville, Virginia region. In 2022, PHA received a Tier III Capacity Building Program Enhancement Grant from Virginia Housing to evaluate and improve their programs aimed at helping first-time homebuyers.

### How PHA helps first-time homebuyers

PHA offers two programs to help buyers earning less than 80% of Area Median Income (AMI) purchase their first home.

1. The **Down Payment Loan** (DPL) program combines funding from several different sources to provide buyers with additional assistance to increase their total down payment and cover closing costs. This allows buyer to more easily afford homes at prices they would otherwise be unable to afford.
2. PHA administers the **Sponsoring Partnerships & Revitalizing Communities** (SPARC) program, which lowers a buyer's mortgage interest rate by 1.00%. SPARC funds are allocated by Virginia Housing and help buyers save thousands of dollars over the life of their loan.

PHA seeks a deeper understanding of these two programs' outcomes and impacts, particularly in terms of racial equity and wealth-building. This document was prepared by HDAdvisors to develop methods for evaluating the program using internal and external data.

## Goals

The primary goal of this analysis is a preliminary evaluation of the DPL and SPARC programs' current abilities to support buyers who are more likely to be at a systemic disadvantage because of unjust lending practices and other discriminatory efforts.

The secondary goal is to provide PHA staff with reproducible methods for collecting, preparing, and analyzing the data necessary for this evaluation. This will allow PHA to regularly track and monitor these programs without outside assistance.

### **i Why this matters**

This analysis will be helpful for PHA in three important ways:

1. To proactively assess whether the program fulfills all applicable fair housing requirements,
2. To provide staff with a thorough understanding of program outcomes for stronger grant applications, and
3. To further PHA's existing Diversity, Equity, and Inclusion (DEI) efforts.

### **Guiding questions**

The following questions help determine (1) what kind of data is needed, and (2) how that data should be analyzed.

- How do PHA's clients vary by race, ethnicity, and income compared to the region as a whole?
- Do the levels of assistance vary significantly by race, ethnicity, and income?
- Are there noticeable migration patterns for clients purchasing their first home?
- How do the homes purchased by clients compare to similar homes in that neighborhood?
- How much home equity have past clients gained? How similar are these gains across race, ethnicity, and neighborhoods?

### **Data assembly**

This section outlines each metric necessary for analysis, along with methods for collecting and cleaning the appropriate data.

#### **Program and client data**

PHA provided data on DPL and SPARC clients from FY 2018 through FY 2023.

This data includes attributes about each client household, including:

- Total annual income
- Area median income (AMI)
- Race and ethnicity (optional)
- Number of persons
- Street address (at time of application)

It also includes DPL-specific program data, including:

- Number of loans provided
- Total amount of DPL assistance
- Funding sources used for each loan
- Source(s) and amount of non-DPL assistance

Information on the homes purchased by clients include:

- Street address
- Sales price
- Appraised value
- Closing costs and other prepaids

Information about clients' mortgages are also provided:

- Lending institution
- Mortgage product
- Mortgage amount
- Interest rate
- Original interest rate (SPARC only)
- Closing date

The full dataset includes multiple other fields that are not necessary or relevant to this analysis.

### Step 1: Import data

```
# Import list of DPL clients and clean column names

clients_dpl_raw <- read_xlsx("data/raw/clients.xlsx",
                             sheet = "Down Payment Loan") |>
  clean_names()

# Import list of SPARC clients and clean column names

clients_sparc_raw <- read_xlsx("data/raw/clients.xlsx",
                                sheet = "SPARC") |>
```

```
clean_names()
```

## Step 2: Clean up and fix data

```
# Clean up DPL data

clients_dpl <- clients_dpl_raw |>

# Drop rows that do not have client records

drop_na(contract_date) |>

# Remove columns with personal information or unnecessary data

select(!c(fha_case_number, ami_approval_date, client_intake_date,
         client_last_name, client_first_name, counselor,
         service_type, client_employer, employer_msa,
         property_msa, status, loan_officer, client_msa,
         client_email_address, phone_number
        )) |>

# Rename certain columns to more helpful descriptions

rename(mortgage_product = x1st_mortgage_product,
       mortgage_amount = x1st_mortgage_amount,
       pha_loans_n = number_of_pha_loans,
       pha_loans_amount = of_pha_loans,
       pha_loan_source = source_of_pha_loan
      ) |>

# Create new unique DPL client ID column

mutate(id_dpl = str_c("DPL", str_pad(row_number(), 3, pad = "0")),
       .before = 1) |>

# Clean up original client_number column

mutate(client_number = str_remove_all(client_number, "\\.0")) |>

# Change "Unknown" values to NA in appraised_value
```

```

mutate(appraised_value = as.numeric(na_if(appraised_value, "Unknown")) ) |>

# Clean up lending_institution names

mutate(lending_institution = case_when(
  str_detect(lending_institution, "Coast") ~ "Atlantic Coast Mortgage",
  str_detect(lending_institution, "Fulton") ~ "Fulton Mortgage",
  str_detect(lending_institution, "Charlottesville") ~ "Habitat for Humanity of Greater
  str_detect(lending_institution, regex("first", ignore_case = T)) ~ "Towne First Mortga
  str_detect(lending_institution, "USDA") ~ "USDA",
  str_detect(lending_institution, "Waterstone") ~ "Waterstone Mortgage Corporation",
  lending_institution == "Towne Bank Mortgage" ~ "TowneBank Mortgage",
  TRUE ~ lending_institution
)) |>

# Clean up mortgage_product names

mutate(mortgage_product = case_when(
  str_detect(mortgage_product, "502|USDA|RHS") ~ "502 Direct Loan",
  mortgage_product == "Coventional" ~ "Conventional",
  str_detect(mortgage_product, "VHDA FHA|VHDA - HFA") ~ "VHDA - FHA",
  TRUE ~ mortgage_product
)) |>

# Change '$-' values in other_assistance_amount to NA

mutate(other_assistance_amount = as.numeric(na_if(other_assistance_amount, "$-")) ) |>

# Create column for single or multiple race; clean up race names

mutate(multirace = case_when(
  str_detect(race, "Multi") ~ "Multiple",
  TRUE ~ "Single"),
  .before = race) |>

mutate(race = case_when(
  str_detect(race, "and White") ~ "Black and White",
  str_detect(race, "Other") ~ "Other races",
  str_detect(race, "Asian") ~ "Asian",
  str_detect(race, "Single Race - Black") ~ "Black",
  TRUE ~ "White"
)

```

```

)) |>

# Simplify rural_area_status and english_proficiency columns

mutate(rural_area_status = case_when(
  str_detect(rural_area_status, "lives") ~ "Rural",
  TRUE ~ "Not rural"
)) |>

mutate(english_proficiency = case_when(
  str_detect(english_proficiency, "is not") ~ "Not proficient",
  TRUE ~ "Proficient"
))

# Clean up SPARC data

clients_sparc <- clients_sparc_raw |>

# Remove columns with personal information or unnecessary data

select(!c(last_name, first_name, client,
          lenders_name, x20)) |>

# Rename certain columns to match DPL data

rename(sale_price = sales_price,
       mortgage_amount = base_loan_amount,
       household_annual_income = borrower_s_income,
       household_size = number_in_hh,
       locality = msa,
       mortgage_product = loan_product,
       lending_institution = lending_company
) |>

# Change client_number column to string

mutate(client_number = as.character(client_number)) |>

# Create new unique SPARC client ID column

mutate(id_sparc = str_c("SPC", str_pad(row_number(), 3, pad = "0"))),

```

```

.before = 1) |>

# Fix interest_rate values

mutate(interest_rate = case_when(
  interest_rate > 1 ~ interest_rate/100,
  TRUE ~ interest_rate
)) |>

# Create mortgage_notes column for specific loan info

mutate(mortgage_notes = case_when(
  str_detect(mortgage_product, "No MI") ~ "No mortgage insurance",
  str_detect(mortgage_product, "Reduced") ~ "Reduced mortgage insurance",
  str_detect(mortgage_product, "Second Plus") ~ "Second Plus",
  TRUE ~ NA_character_),
  .after = mortgage_product
) |>

# Clean up mortgage_product names to match DPL data

mutate(mortgage_product = case_when(
  str_detect(mortgage_product, regex("Conventional", ignore_case = T)) ~ "Conventional",
  str_detect(mortgage_product, regex("RHS", ignore_case = T)) ~ "502 Direct Loan",
  str_detect(mortgage_product, regex("FHA", ignore_case = T)) ~ "FHA",
  is.na(mortgage_product) ~ "Unknown",
  TRUE ~ mortgage_product
)) |>

# Clean up lending_institution names to match DPL data

mutate(lending_institution = case_when(
  str_detect(lending_institution, "C&F") ~ "C&F Mortgage Corporation",
  str_detect(lending_institution, "Waterstone") ~ "Waterstone Mortgage Corporation",
  is.na(lending_institution) ~ "Unknown",
  TRUE ~ lending_institution
)) |>

# Clean up yes/no columns

mutate(received_cd = case_when(

```

```

    str_detect(received_cd, "YES|Yes|yes") ~ "Yes",
    received_cd == "no" ~ "No",
    received_cd == "N/A" ~ NA_character_,
    TRUE ~ NA_character_
)) |>

mutate(in_ors = case_when(
  str_detect(in_ors, "No -") ~ "No",
  TRUE ~ in_ors
))

```

### Step 3: Join DPL and SPARC client data

Some clients use both the DPL and SPARC programs to purchase their home. For now, these clients were identified by manually cross-referencing names and other fields. To join the data, the DPL `client_number` values were assigned to the respective client records in the SPARC data.

In the future, PHA should maintain a single client dataset that can differentiate between DPL-only, SPARC-only, and “double-dipping” clients. Specific recommendations will be detailed at the end of this document.

```

# Create full join using client_number

clients_all <- clients_dpl |>
  full_join(clients_sparc, by = "client_number", na_matches = "never",
            suffix = c("_dpl", "_spc")) |>

# Reorder columns to place duplicate fields together

select(2, 1, 46, 3, 4, 47, 5, 52, 6, 7, 53, 8, 9,
       10, 11, 12, 13, 14, 50, 15, 16, 17, 59, 18,
       55, 19, 51, 20, 49, 48, 50, 21, 22, 23, 24,
       25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
       36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 54,
       56, 57, 60, 61
) |>

# Create column to show whether client uses one or both programs

mutate(program = case_when(
  !is.na(id_dpl) & !is.na(id_sparc) ~ "Both",

```

```

!is.na(id_dpl) & is.na(id_sparc) ~ "DPL",
TRUE ~ "SPARC"),
.before = 1) |>

# Create universal client ID column

mutate(id_pha = str_c("PHA", str_pad(row_number(), 3, pad = "0")),
.before = 1)

```

#### Step 4: View data

Cleaned and joined client data are shown in the table below.

```

reactable(clients_all,
          defaultPageSize = 5,
          defaultColDef = colDef(minWidth = 150),
          compact = TRUE,
          striped = TRUE,
          wrap = FALSE)

```

id_pha	program	client_number	id_dpl	id_sparc	contract_date	close_date_dpl	close_date_spc	household_annua...	household_annua...
PHA001	Both	4305081	DPL001	SPC002	2022-05-24T00:00...	2022-07-26T00:00...	2022-07-27T00:00...	59600.08	66000.08
PHA002	DPL	4658207	DPL002		2022-06-25T00:00...	2022-08-02T00:00...		58323.44	
PHA003	DPL	4650788	DPL003		2021-07-21T00:00...	2021-08-21T00:00...		55016	
PHA004	Both	4721465	DPL004	SPC022	2021-06-28T00:00...	2021-08-21T00:00...	2021-08-27T00:00...	35459.84	33234
5 rows									

1-5 of 131 rows

Previous 1 2 3 4 5 ... 27 Next

Where there were duplicate fields from each dataset, the \_dpl and \_spc suffixes were added to the end of column names, respective to their source. For example, the close date data from the DPL records are retained as the `close_date_dpl` column, while the close dates from the SPARC data are `close_date_spc`.

#### Data integrity issues

Both the DPL and SPARC datasets were found to have data integrity issues that PHA should resolve in the future.

- Mortgage product fields included both “RHS” (Rural Housing Service), “USDA”, and “502” values. For now, these are all assumed to be “502 Direct Loans”.
- Overlapping fields from the DPL and SPARC datasets do not always reconcile for the clients who used both programs. For example, the specific close dates and household income values are often slightly different—but should, in theory, be identical.
- The SPARC client records are missing many values in the closing date, interest rate, household income, household size, and lender columns.
- Some DPL clients were noted as receiving SPARC in the `other_assistance` column. However, two of clients were *not* found in the SPARC data.

## Regional and neighborhood profiles

#### Why this is needed

Demographic and socioeconomic data will help establish the baseline characteristics of population groups that PHA seeks to compare with its clients. For example, how closely does the client pool match the average racial and ethnic makeup across the region?

Two datasets will be used to create summary profiles of the geographies PHA will use as reference for its clients’ characteristics. Both are from the U.S. Census Bureau:

- 2020 Decennial Census
- American Community Survey (5-year estimates)

#### Note

As of January 2023, the latest ACS data available are the 2017-2021 5-year estimates.

The fields to collect from this data include:

- Race and ethnicity of population
- Average household incomes
- Homeownership rates

### Step 1: Race and ethnicity data from 2020 Census

```
# Get variables for 2020 Census PL-94171 Redistricting dataset

pl_vars <- load_variables(2020, "pl")

# Find and select race and ethnicity variables

race_vars <- pl_vars |>
  filter(name %in% c(
    "P1_003N", # White
    "P1_004N", # Black
    "P1_005N", # American Indian and Alaska Native
    "P1_006N", # Asian
    "P1_007N", # Native Hawaiian and Other Pacific Islander
    "P1_008N", # Some other race
    "P1_009N", # Two or more races
    "P2_002N", # Total Hispanic or Latino
    "P2_003N" # Total not Hispanic or Latino
  )) |>

# Rename race and ethnicity groups

mutate(label = case_when(
  name == "P1_003N" ~ "White",
  name == "P1_004N" ~ "Black",
  name == "P1_005N" ~ "Another race",
  name == "P1_006N" ~ "Asian",
  name == "P1_007N" ~ "Another race",
  name == "P1_008N" ~ "Another race",
  name == "P1_009N" ~ "Multiracial",
  name == "P2_002N" ~ "Hispanic or Latino",
  name == "P2_003N" ~ "Not Hispanic or Latino",
)) |>

# Add race/ethnicity label
```

```

    mutate(category = case_when(
      str_detect(name, "P1") ~ "Race",
      TRUE ~ "Ethnicity"),
      .after = 1
    ) |>

    # Drop concept column (unneeded)

    select(-concept)

  # Import 2020 Census race and ethnicity data for Charlottesville City

  cv_race_raw <- get_decennial(
    geography = "county",
    variables = race_vars$name,
    year = 2020,
    sumfile = "pl",
    state = "VA",
    county = "Charlottesville city"
  )

  # Join race and ethnicity labels

  cv_race <- cv_race_raw |>
    left_join(race_vars, by = c("variable" = "name")) |>

    # Sum "Another race" values into one row

    group_by(category, label) |>
    summarise(value = sum(value)) |>

    # Add percent column

    group_by(category) |>
    mutate(pct = value/sum(value)) |>
    ungroup()

  # Generate table

  cv_race |>
    arrange(desc(value), .by_group = TRUE) |>

```

category	label	value	pct
Ethnicity	Not Hispanic or Latino	43346	0.9311108
Race	White	30344	0.6518162
Race	Black	7122	0.1529869
Race	Asian	4083	0.0877065
Race	Multiracial	3583	0.0769660
Ethnicity	Hispanic or Latino	3207	0.0688892
Race	Another race	1421	0.0305243

```
kable() |>
kableExtra::kable_styling(
  bootstrap_options = c("striped", "hover", "condensed", "responsive")
)
```

## Step 2: Household income data from ACS

```
# Get variables for Table B25118: Tenure by Household Income

b25118_vars <- load_variables(2021, "acs5") |>
  filter(str_sub(name, end = 6) %in% "B25118")

# Clean variables

inc_vars <- b25118_vars |>
  separate(label, c("est", "total", "tenure", "income"), sep = "!!!") |>
  select(1, 4, 5) |>
  drop_na() |>
  mutate(tenure = str_remove_all(tenure, " occupied:"),
         tenure = str_replace_all(tenure, "Owner", "Homeowner"))

# Import 2021 ACS estimates

b25118_raw <- get_acs(
  geography = "county",
  variables = inc_vars$name,
  year = 2021,
  county = "Charlottesville city",
  state = "VA"
)
```

```

# Join variables to data

cv_income <- b25118_raw |>
  left_join(inc_vars, by = c("variable" = "name")) |>

# Remove unnecessary columns and rearrange

select(6, 7, 4) |>

# Collapse into fewer income categories

mutate(income = case_when(
  income == "Less than $5,000" ~ "Less than $35,000",
  income == "$5,000 to $9,999" ~ "Less than $35,000",
  income == "$10,000 to $14,999" ~ "Less than $35,000",
  income == "$15,000 to $19,999" ~ "Less than $35,000",
  income == "$20,000 to $24,999" ~ "Less than $35,000",
  income == "$25,000 to $34,999" ~ "Less than $35,000",
  income == "$100,000 to $149,999" ~ "$100,000 or more",
  income == "$150,000 or more" ~ "$100,000 or more",
  TRUE ~ income
)) |>
  group_by(tenure, income) |>
  summarise(estimate = sum(estimate))

# Generate table

cv_income |>
  ungroup() |>
  mutate(income = fct_relevel(income, "Less than $35,000", "$35,000 to $49,999",
                               "$50,000 to $74,999", "$75,000 to $99,999",
                               "$100,000 or more")) |>
  arrange(income) |>
  kable() |>
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "hover", "condensed", "responsive")
)

```

### Step 3: Homeownership data from ACS

```
# tbd
```

tenure	income	estimate
Homeowner	Less than \$35,000	883
Renter	Less than \$35,000	5140
Homeowner	\$35,000 to \$49,999	764
Renter	\$35,000 to \$49,999	1234
Homeowner	\$50,000 to \$74,999	861
Renter	\$50,000 to \$74,999	1809
Homeowner	\$75,000 to \$99,999	1171
Renter	\$75,000 to \$99,999	985
Homeowner	\$100,000 or more	4300
Renter	\$100,000 or more	2165

# tbd

## Mortgage applications and activity

### 💡 Why this is needed

Along with comparing client characteristics with the region or a neighborhood as a whole, we can compare their mortgages to overall lending activity in the community over time. This will allow PHA to benchmark its buyer assistance efforts to home loan trends across the region.

This information is available from the Home Mortgage Disclosure Act (HMDA) data from the Consumer Financial Protection Bureau. HMDA data include loan-level information on buyer demographics, loan attributes, and property characteristics. Entries for buyers whose applications were eventually withdrawn or denied are also included.

### ℹ Note

As of January 2023, the latest HMDA data available is for 2021.

The fields to collect from this data include:

- Buyer race and ethnicity
- Buyer county and census tract
- Loan type (conventional, FHA, VA, or USDA)
- Action taken (loan originated, application denied or withdrawn)
- Loan amount
- Interest rate

- Property value
- Occupancy type (principal residence only)
- Buyer gross annual income
- Debt-to-income ratio

Add steps on cleaning/preparing this data

## Home values

### Why this is needed

Detailed data on home sales and values are necessary to make estimates about the total home equity gained (or lost) by PHA buyers and other homeowners in the community. This property-level data will help PHA assess its programs' abilities to support wealth-building among clients.

## Real estate assessments

Localities in PHA's service area regularly assess property values to calculate the total real estate taxes to be paid by each property owner. Properties are reassessed to better match taxable values with potential market values, which can change significantly over time.

Despite these updates, local assessments can often lag behind current market trends, so they are not the most accurate indicator of the actual market value of a home if it were sold. Still, assessment data are consistent and publicly available for all parcels in a locality. This creates a useful longitudinal dataset.

For this preliminary analysis, only assessment data from the City of Charlottesville will be used.

The fields to collect from this data include:

- Parcel number
- Street number and name
- Bedrooms and bathrooms
- Finished square footage
- Total assessed value

Add steps on cleaning/preparing this data

## **MLS sales**

Actual recorded home sales are the most accurate way to determine residential property values. For arms-length transactions, sales prices reflect the actual market value of each home. These records are therefore the ideal source to best determine home equity changes among homeowners.

PHA is able to provide data exported from the multiple listing service (MLS) operated by the Charlottesville Area Association of REALTORS® (CAAR). The MLS platform includes all real estate listings from agents in the region and is dynamically updated as listings are added, updated, or closed.

For this preliminary analysis, PHA has provided data on all sold single-family homes in the City of Charlottesville from September 2022 through November 22, 2022.

Fields included in this exports include:

- Property address
- Days on market
- List price
- Closed date
- Bedrooms and bathrooms
- Finished square footage

Add steps on cleaning/preparing this data

## **Other data**

May or may not eventually be used

- Neighborhood boundaries from City of Charlottesville ([link](#))
- Real estate sales ([link](#))

## **Analysis**

### **Clients served by program**

From July 2017 through September 2022, PHA served a total of 131 households with its DPL and SPARC programs.

```

# Cumulative total served by program

clients_total <- clients_all |>
  select(program, close_date_dpl, close_date_spc) |>

  # Collapse close dates into one column
  # Assume DPL close date is correct when SPARC date differs

  mutate(close_date = case_when(
    !is.na(close_date_dpl) ~ close_date_dpl,
    TRUE ~ close_date_spc
  ))
}

# Summarise clients by month

clients_cum <- clients_total |>
  drop_na(close_date) |>
  count(program, month = floor_date(close_date, "month")) |>
  ungroup() |>
  complete(program, month, fill = list(n = 0)) |>

  # Cumulative sum by program

  group_by(program) |>
  mutate(cum_sum = cumsum(n))

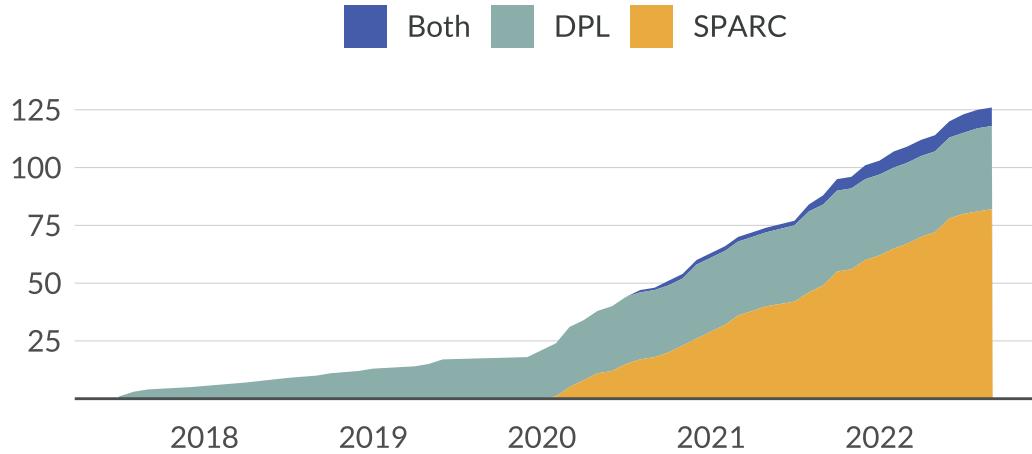
# Create plot

ggplot(clients_cum, aes(x = month, y = cum_sum, fill = program)) +
  geom_area() +
  scale_y_continuous(breaks = c(25,50,75,100,125)) +
  scale_fill_hda() +
  labs(title = "Cumulative total of clients served by program",
       subtitle = "Data from July 2017 through September 2022",
       caption = "***Note:** Four records without close dates are omitted.") +
  theme_hda() +
  add_zero_line("y") +
  theme(
    legend.position = "top"
  )

```

## Cumulative total of clients served by program

Data from July 2017 through September 2022



**Note:** Four records without close dates are omitted.

PHA served DPL-only clients from 2017 through 2019, averaging fewer than ten annually. In 2020, PHA began offering SPARC, which has been used by more than 25 clients annually since. Several clients have used both DPL and SPARC assistance each year since 2020.

```
# Summarise data by program and year

clients_annual <- clients_total |>
  drop_na(close_date) |>
  count(program, year = floor_date(close_date, "year")) |>
  ungroup() |>
  complete(program, year, fill = list(n = 0)) |>
  mutate(year = format(year, "%Y"))

# Create plot

ggplot(clients_annual, aes(x = year, y = n, fill = program)) +
  geom_col(position = "stack") +
  scale_fill_hda() +
  labs(title = "Annual number of clients served by program",
       subtitle = "Data from July 2017 through September 2022",
       caption = "**Note:** Four records without close dates are omitted.") +
  theme_hda()
```

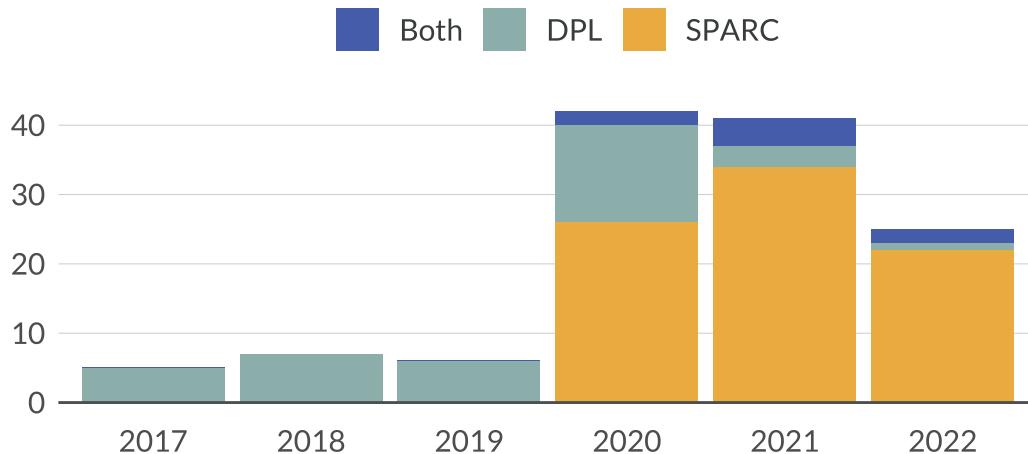
```

add_zero_line("y") +
theme(
  legend.position = "top"
)

```

## Annual number of clients served by program

Data from July 2017 through September 2022



**Note:** Four records without close dates are omitted.

In total, two-thirds (66 percent) of all clients used SPARC only. About one quarter (27 percent) used DPL only, and just eight clients (6 percent) took advantage of both.

```

# Summarise data by program

clients_program <- clients_total |>
  count(program)

# Create plot

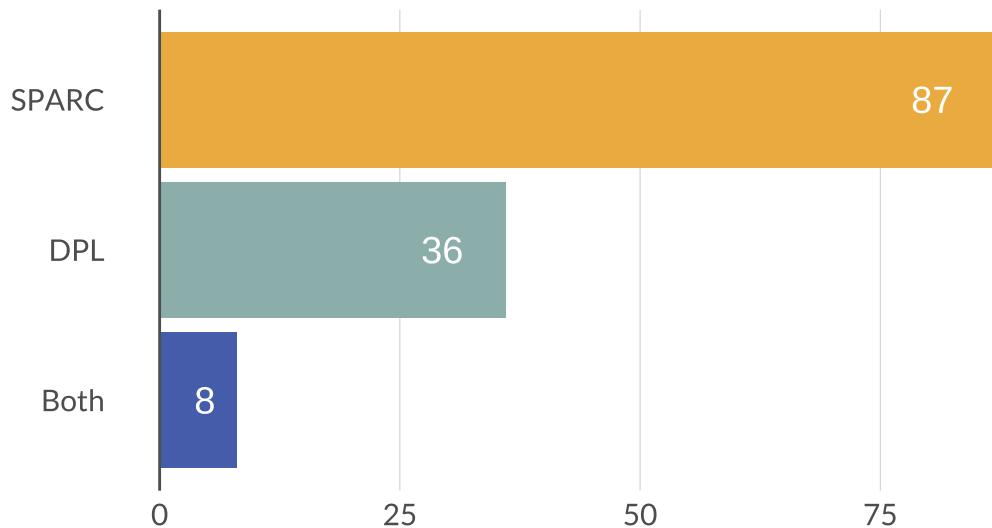
ggplot(clients_program, aes(y = reorder(program, n), x = n, fill = program, label = n)) +
  geom_col() +
  geom_text(hjust = 2,
            color = "white",
            size = 5) +
  scale_fill_hda() +

```

```
labs(title = "Total number of clients served by program",
     subtitle = "Data from July 2017 through September 2022") +
theme_hda() +
add_zero_line("x") +
flip_gridlines()
```

## Total number of clients served by program

Data from July 2017 through September 2022



### Profile of clients

#### ⚠ Note

Race and ethnicity data is only available for clients who used the DPL program.

Among the 44 clients who used the DPL program, most were white (41 percent) or Black (34 percent). Another 11 percent were Asian, with the remainder being multiracial or of another race. This is a more diverse pattern than Charlottesville as a whole, where 65 percent of residents are white, while only 15 percent of residents are Black, and 9 percent are Asian.

The share of DPL clients who are Hispanic or Latino is about 7 percent, which aligns very closely with the overall population of Charlottesville.

```

# Summarise clients by race

clients_race <- clients_all |>
  count(race) |>
  drop_na() |>
  ungroup() |>

# Add percent column

mutate(pct = n/sum(n)) |>

# Add source and category columns

mutate(source = "Clients",
       category = "Race",
       .before = 1) |>

# Update race labels and column names to match ACS data

mutate(race = case_when(
  race == "Black and White" ~ "Multiracial",
  race == "Other races" ~ "Another race",
  TRUE ~ race)) |>
  rename(label = race,
         value = n)

# Same as above for ethnicity

clients_ethnicity <- clients_all |>
  count(ethnicity) |>
  drop_na() |>
  ungroup() |>

# Add percent column

mutate(pct = n/sum(n)) |>

# Add source and category columns

mutate(source = "Clients",
       category = "Ethnicity",

```

```

.before = 1) |>

# Update race labels and column names to match ACS data

mutate(ethnicity = case_when(
  ethnicity == "Hispanic" ~ "Hispanic or Latino",
  ethnicity == "Not Hispanic" ~ "Not Hispanic or Latino",
  TRUE ~ ethnicity)) |>
  rename(label = ethnicity,
         value = n)

# Add client data to ACS data

race_join <- cv_race |>
  mutate(source = "Charlottesville", .before = 1) |>
  bind_rows(clients_race, clients_ethnicity)

race_plot <- race_join |>
  filter(category == "Race") |>
  ggplot(aes(x = pct, y = source, fill = reorder(label, pct), alpha = source)) +
    geom_col(position = "stack") +
    scale_fill_hda(-1) +
    scale_alpha_discrete(range = c(0.6, 1)) +
    scale_x_continuous(labels = label_percent()) +
    guides(fill = guide_legend(reverse = TRUE),
           alpha = "none") +
    labs(title = "Client race and ethnicity compared to Charlottesville",
         subtitle = "Includes only DPL clients",
         fill = "Race") +
    theme_hda() +
    flip_gridlines() +
    theme(
      legend.position = "left",
      legend.justification = "left",
      legend.title = element_text(hjust = 0)
    )

ethnicity_plot <- race_join |>
  filter(category == "Ethnicity") |>
  ggplot(aes(x = pct, y = source, fill = reorder(label, pct), alpha = source)) +
    geom_col(position = "stack") +

```

```

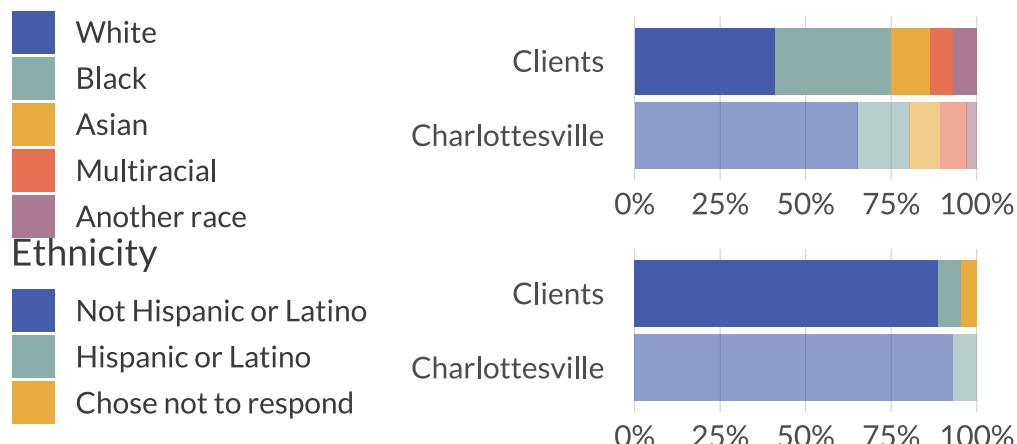
scale_fill_hda(-1) +
  scale_alpha_discrete(range = c(0.6, 1)) +
  scale_x_continuous(labels = label_percent()) +
  guides(fill = guide_legend(reverse = TRUE),
         alpha = "none") +
  labs(fill = "Ethnicity",
       caption = "**Source:** U.S. Census Bureau, 2020 Decennial Census P.L. 94-171 Redi-
theme_hda() +
  flip_gridlines() +
  theme(
    legend.position = "left",
    legend.justification = "left",
    legend.title = element_text(hjust = 0)
  )

race_plot + ethnicity_plot +
plot_layout(ncol = 1)

```

## Client race and ethnicity compared to Charlottesville

Includes only DPL clients



Source: U.S. Census Bureau, 2020 Decennial Census P.L. 94-171 Redistricting

Household incomes are available for 88 of the 131 clients. Of the 43 clients with no income data, all are SPARC-only records. Six of the eight DPL clients who also used SPARC have two incomes recorded for each program—these values all differ, sometimes significantly.

```

# Summarise clients by household income

clients_inc <- clients_all |>

# Select only id_pha, program, and income columns

select(1, 2, 9, 10, 11) |>

# Combine DLP and SPARC incomes into one column

mutate(income = case_when(
  !is.na(household_annual_income_dpl) & !is.na(household_annual_income_spc) ~ household_
  !is.na(household_annual_income_dpl) & is.na(household_annual_income_spc) ~ household_a
  is.na(household_annual_income_dpl) & !is.na(household_annual_income_spc) ~ household_a
  TRUE ~ NA_real_
)) |>

# Add column describing income data availability

mutate(data = case_when(
  !is.na(household_annual_income_dpl) & !is.na(household_annual_income_spc) ~ "Different"
  !is.na(household_annual_income_dpl) & is.na(household_annual_income_spc) ~ "Income rec
  is.na(household_annual_income_dpl) & !is.na(household_annual_income_spc) ~ "Income rec
  TRUE ~ "No data"
))

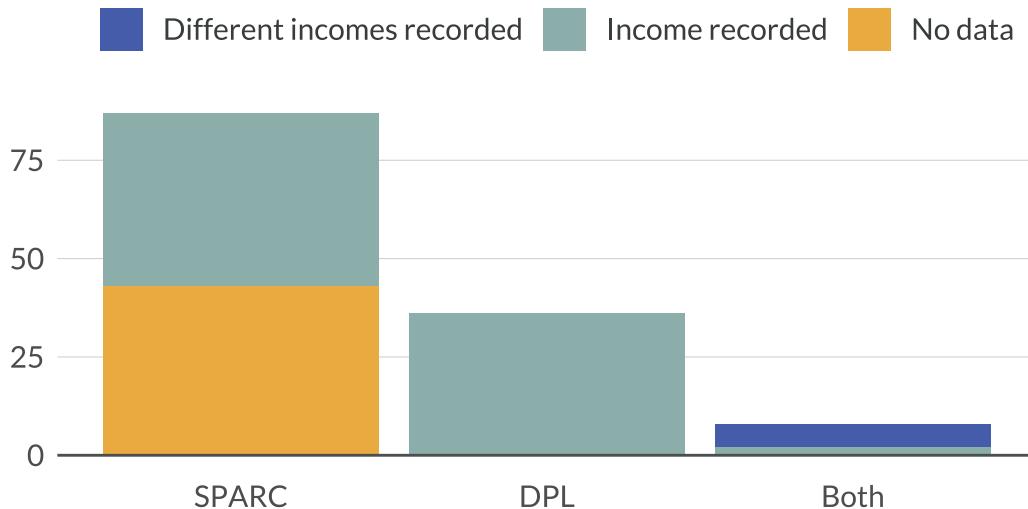
# Create plot showing income data availability

clients_inc |>
  group_by(program, data) |>
  summarise(n = n()) |>
  ggplot(aes(x = reorder(program, n, decreasing = TRUE), y = n, fill = data)) +
    geom_col(position = "stack") +
    scale_fill_hda() +
    labs(title = "Client income data availability by program",
         subtitle = "Data from July 2017 through September 2022") +
    theme_hda() +
    add_zero_line("y") +
    theme(
      legend.position = "top"
)

```

## Client income data availability by program

Data from July 2017 through September 2022



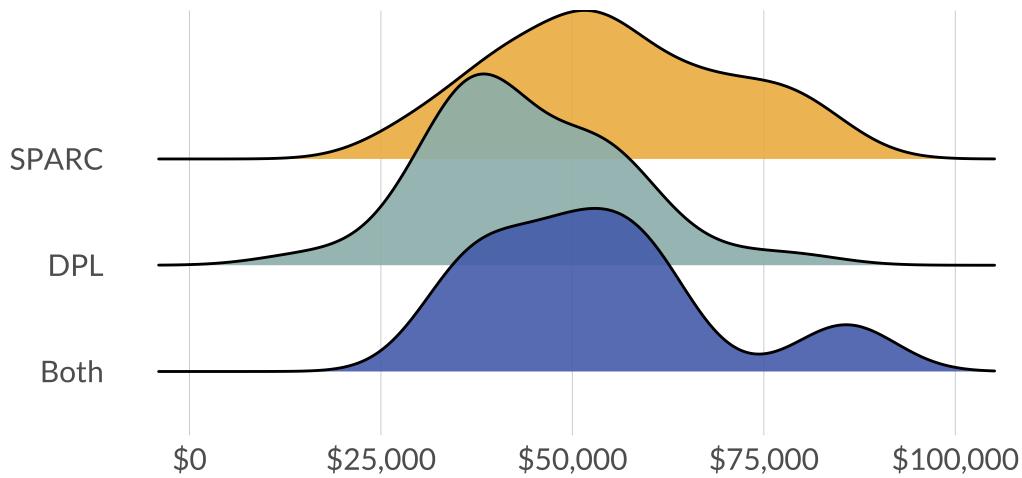
Most of PHA's clients have household incomes between \$25,000 and \$60,000. However, the incomes of SPARC-only clients skew slightly higher, with some above \$75,000.

```
# Summarise income by program

ggplot(clients_inc, aes(x = income, y = program, fill = program)) +
  geom_density_ridges(alpha = 0.9) +
  scale_fill_hda() +
  scale_x_continuous(labels = label_dollar()) +
  labs(title = "Client household income by program",
       subtitle = "Data from July 2017 through September 2022",
       caption = "**Note:** Does not include 43 SPARC clients with no recorded income.") +
  theme_hda() +
  flip_gridlines()
```

## Client household income by program

Data from July 2017 through September 2022



**Note:** Does not include 43 SPARC clients with no recorded income.

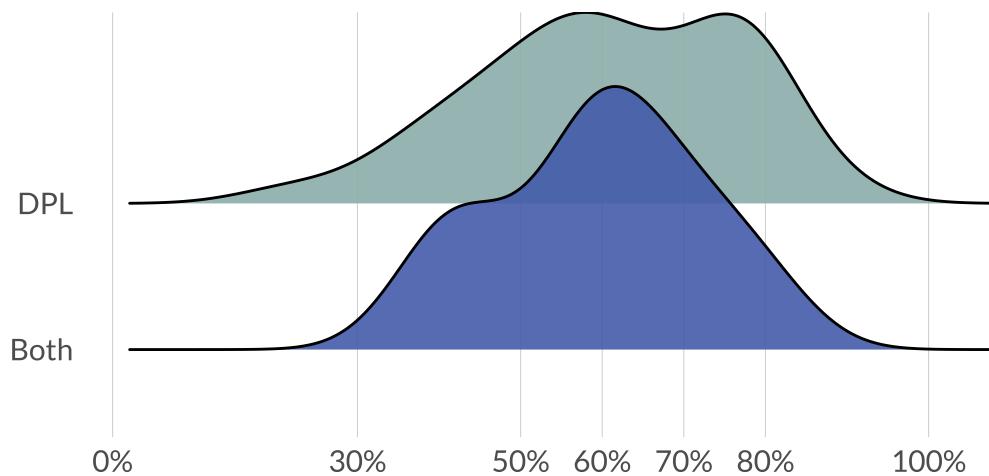
Area median incomes (AMI) are calculated only for clients who used the DPL program. AMIs are most commonly between 50 and 80 percent, in line with program eligibility guidelines. On average, AMIs of the DPL clients who also used SPARC are slightly lower than those of clients who used DPL only.

```
# Summarise AMI by program

clients_inc |> filter(program != "SPARC") |>
  ggplot(aes(x = client_ami, y = program, fill = program)) +
  geom_density_ridges(alpha = 0.9) +
  scale_fill_hda() +
  scale_x_continuous(labels = label_percent(), breaks = c(0, 0.3, 0.5, 0.6, 0.7, 0.8, 1))
  labs(title = "Client AMI by program",
       subtitle = "Data from July 2017 through September 2022",
       caption = "**Note:** Does not include 43 SPARC clients with no recorded income.") +
  theme_hda() +
  flip_gridlines()
```

## Client AMI by program

Data from July 2017 through September 2022



**Note:** Does not include 43 SPARC clients with no recorded income.

Compared to the distribution of household incomes in Charlottesville, clients are much more likely to earn more than most renters, while still making significantly less than most homeowners. Over 78 percent of clients earn between \$35,000 and \$75,000, versus just 27 percent of renters and 20 percent of homeowners.

```
# Assign client incomes to ACS categories

clients_inc_sum <- clients_inc |>
  mutate(income = case_when(
    income < 35000 ~ "Less than $35,000",
    income >= 35000 & income < 50000 ~ "$35,000 to $49,999",
    income >= 50000 & income < 75000 ~ "$50,000 to $74,999",
    income >= 75000 & income < 100000 ~ "$75,000 to $99,999",
    income >= 100000 ~ "$100,000 or more",
    TRUE ~ "No data"
  )) |>

# Filter out clients with no income data (optional)

filter(income != "No data") |>

# Summarise clients by category
```

```

count(income) |>

# Get column names to match ACS data

mutate(tenure = "Clients", .before = 1) |>
rename(estimate = n)

# Join ACS income data with client income data

clients_cv_inc <- clients_inc_sum |>
bind_rows(cv_income) |>

# Add percentages

group_by(tenure) |>
mutate(pct = estimate/sum(estimate)) |>

# Reorder income categories

mutate(income = fct_relevel(
  income, "Less than $35,000", "$35,000 to $49,999", "$50,000 to $74,999",
  "$75,000 to $99,999", "$100,000 or more"))

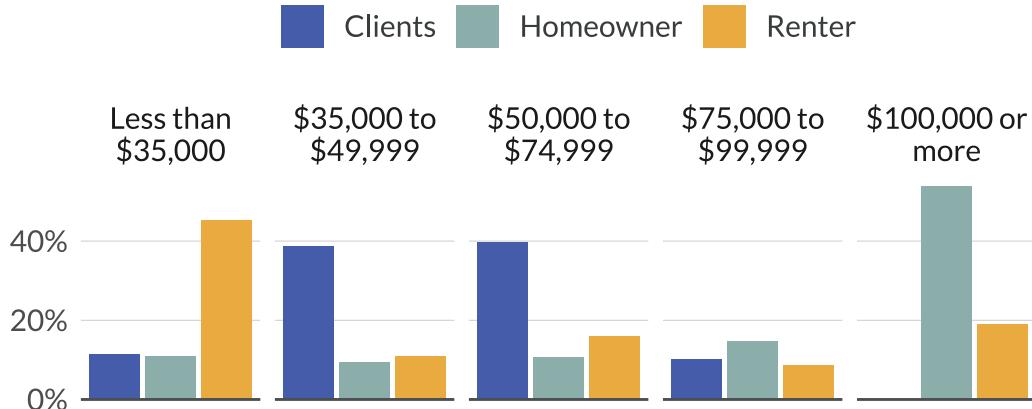
# Create plot

ggplot(clients_cv_inc, aes(y = pct, x = tenure, fill = tenure)) +
  geom_col() +
  facet_wrap(~income, nrow = 1, labeller = label_wrap_gen(12)) +
  scale_y_continuous(labels = label_percent()) +
  scale_fill_hda() +
  labs(title = "Client income compared to Charlottesville",
       subtitle = "Data from July 2017 through September 2022",
       caption = "**Source:** U.S. Census Bureau, American Community Survey, 2017-2021 5-year estimates") +
  theme_hda() +
  add_zero_line("y") +
  theme(axis.text.x = element_blank(),
        legend.position = "top")

```

# Client income compared to Charlottesville

Data from July 2017 through September 2022



Source: U.S. Census Bureau, American Community Survey, 2017-2021 5-year estimates.

Note: Does not include 43 records where client income data not available.

## Assistance levels

PHA's DPL program uses funds from a number of different sources. Clients were most commonly supported by HOME funds from DHCD (18) and from Albemarle County's Homebuyer Assistance Program (11). Overall, PHA has used eight separate sources to fund its DPL program.

```
dpl_sources <- clients_all |>
  filter(program != "SPARC") |>
  select(1, 2, 36:43) |>
  pivot_longer(
    cols = c(3:10),
    names_to = "source",
    values_to = "x"
  ) |>
  group_by(source) |>
  summarise(count = sum(x)) |>
  mutate(source = case_when(
    source == "dhcd" ~ "DHCD HOME",
    source == "achap" ~ "Albemarle County HAP",
    source == "cdfi" ~ "CDFI Regional Down-payment Loan Program",
    source == "sparc" ~ "SPARC"
  )) |>
  mutate(count = ifelse(source == "sparc", 1, count)) |>
  mutate(source_label = case_when(
    source == "dhcd" ~ "DHCD HOME",
    source == "achap" ~ "Albemarle County HAP",
    source == "cdfi" ~ "CDFI Regional Down-payment Loan Program",
    source == "sparc" ~ "SPARC"
  ))
```

```

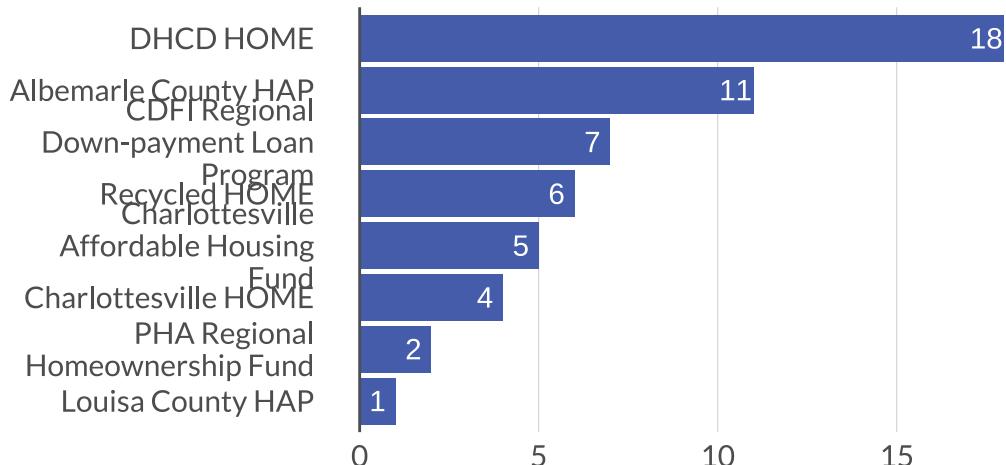
source == "recycled_home" ~ "Recycled HOME",
source == "cahf" ~ "Charlottesville Affordable Housing Fund",
source == "city" ~ "Charlottesville HOME",
source == "pha_regional" ~ "PHA Regional Homeownership Fund",
source == "lchap" ~ "Louisa County HAP",
TRUE ~ str_to_upper(source)
)) |>
mutate(source = str_wrap(source, 20))

ggplot(dpl_sources, aes(x = count, y = reorder(source, count), label = count)) +
geom_col(fill = "#445ca9") +
geom_text(color = "#FFFFFF", nudge_x = -0.5) +
theme_hda() +
flip_gridlines() +
add_zero_line("x") +
labs(
  title = "Number of DPL awards by source",
  subtitle = "Data from July 2017 through September 2022",
  caption = "***Notes:** 'HAP' = Homebuyer Assistance Program."
)

```

## Number of DPL awards by source

Data from July 2017 through September 2022



**Notes:** 'HAP' = Homebuyer Assistance Program.

PHA awarded a total of \$1,242,190 in DPL assistance from July 2017 to September 2022.

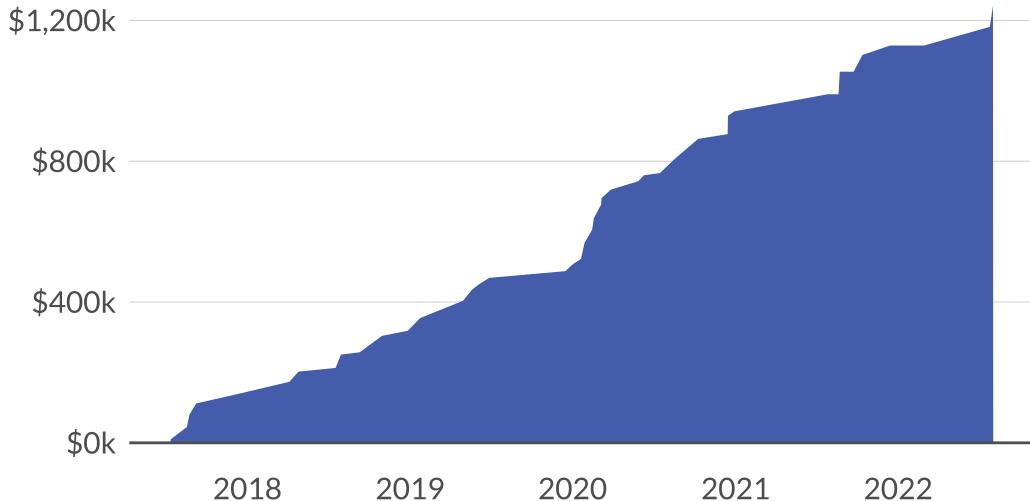
This amounted to roughly \$28,230 provided per client.

```
dpl_amount <- clients_all |>
  drop_na("close_date_dpl") |>
  arrange(close_date_dpl) |>
  mutate(cum_sum = cumsum(pha_loans_amount))

ggplot(dpl_amount, aes(x = close_date_dpl, y = cum_sum)) +
  geom_area(fill = "#445ca9") +
  scale_y_continuous(labels = label_dollar(scale = 0.001, suffix = "k")) +
  add_zero_line("y") +
  theme_hda() +
  labs(
    title = "Total amount of DPL assistance awarded",
    subtitle = "Data from July 2017 through September 2022"
)
```

## Total amount of DPL assistance awarded

Data from July 2017 through September 2022



The average amount each client received stayed around \$28,000 from 2017 through 2020. In 2021, that average increased to \$37,435. In 2022, the two clients that received DPL each received significant DPL assistance well above prior year averages.

```

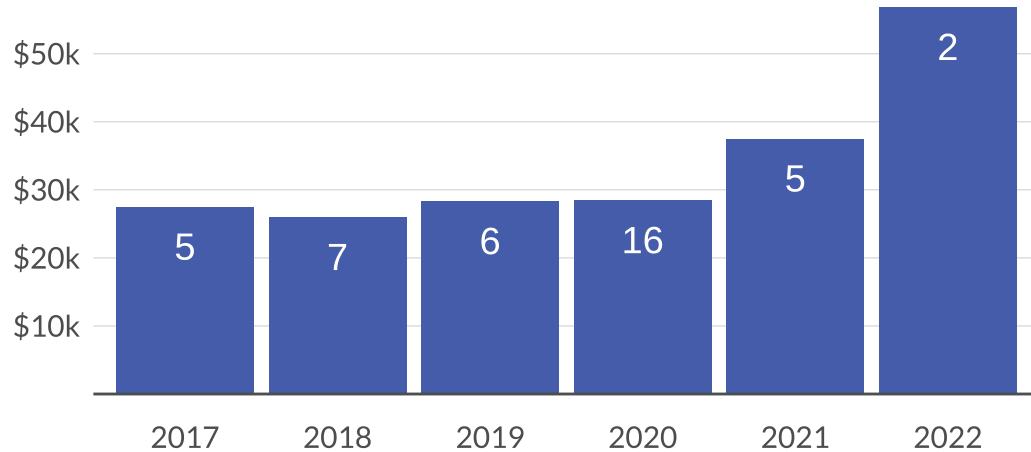
dpl_annual <- dpl_amount |>
  mutate(year = format(close_date_dpl, format = "%Y")) |>
  filter(pha_loans_amount != 0) |>
  group_by(year) |>
  summarise(avg = mean(pha_loans_amount),
            clients = n())

ggplot(dpl_annual, aes(x = year, y = avg, label = clients)) +
  geom_col(fill = "#445ca9") +
  geom_text(vjust = 2,
            color = "white",
            size = 5) +
  scale_y_continuous(labels = label_dollar(scale = 0.001, suffix = "k"),
                     breaks = c(10000, 20000, 30000, 40000, 50000)) +
  add_zero_line("y") +
  theme_hda() +
  labs(
    title = "Average amount of DPL assistance awarded per client",
    subtitle = "Data from July 2017 through September 2022",
    caption = "**Note:** Bars show total number of clients for that group."
)

```

## Average amount of DPL assistance awarded per client

Data from July 2017 through September 2022



**Note:** Bars show total number of clients for that group.

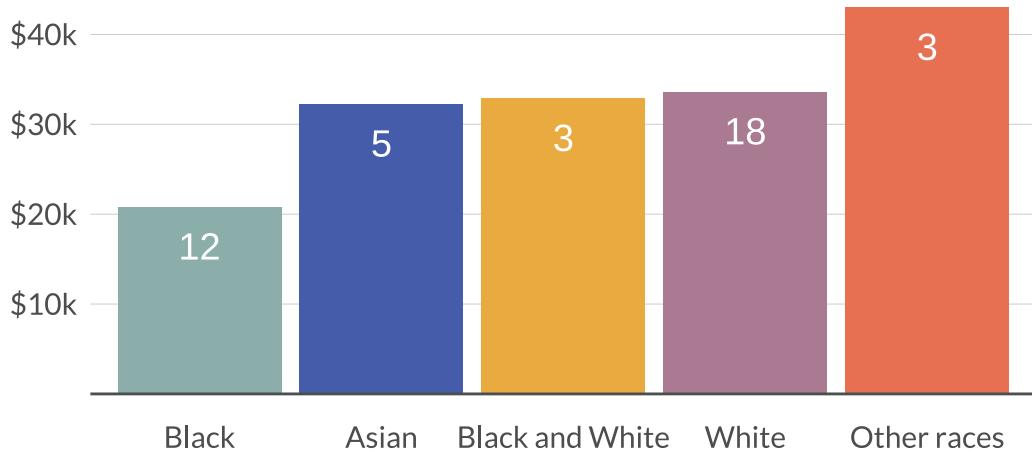
Black clients were the only race with an average DPL amount (\$20,761) below the overall average. Asian, Black and White, and White-alone clients all had averages between \$32,000 and \$34,000. Three clients of another race tallied a notably higher average at \$42,973.

```
dpl_race <- dpl_amount |>
  filter(pna_loans_amount != 0) |>
  group_by(race) |>
  summarise(avg = mean(pna_loans_amount),
            clients = n())

ggplot(dpl_race, aes(x = reorder(race, avg), y = avg, fill = race, label = clients)) +
  geom_col() +
  geom_text(vjust = 2,
            color = "white",
            size = 5) +
  scale_fill_hda() +
  scale_y_continuous(labels = label_dollar(scale = 0.001, suffix = "k"),
                     breaks = c(10000, 20000, 30000, 40000, 50000)) +
  add_zero_line("y") +
  theme_hda() +
  labs(
    title = "Average amount of DPL assistance awarded by race",
    subtitle = "Data from July 2017 through September 2022",
    caption = "**Note:** Bars show total number of clients for that group."
  )
```

## Average amount of DPL assistance awarded by

Data from July 2017 through September 2022



**Note:** Bars show total number of clients for that group.

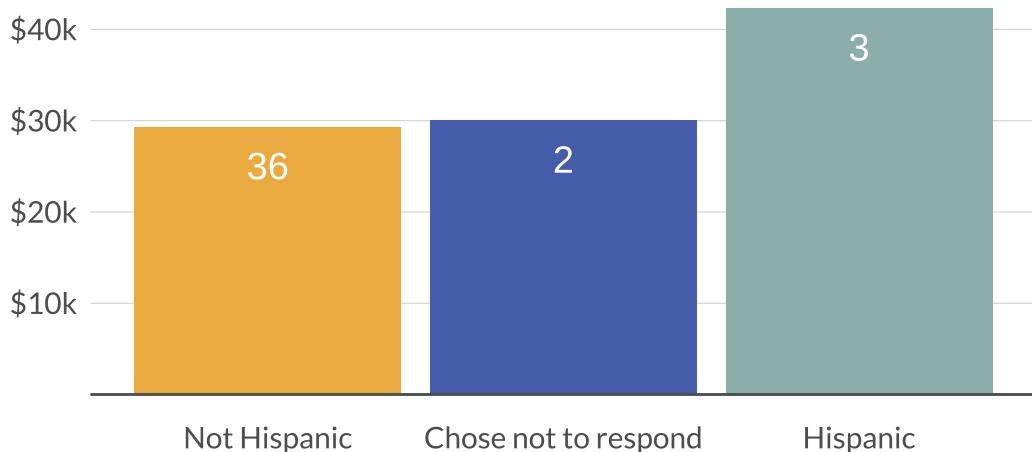
Only five total clients were Hispanic or chose not to respond when asked their ethnicity. As a result, the average DPL amount for non-Hispanic clients (\$29,310) is very close to the overall client average. Among the three clients who were Hispanic, their average DPL amounted to a much larger \$42,316.

```
dpl_ethnicity <- dpl_amount |>
  filter(pha_loans_amount != 0) |>
  group_by(ethnicity) |>
  summarise(avg = mean(pha_loans_amount),
            clients = n())

ggplot(dpl_ethnicity, aes(x = reorder(ethnicity, avg), y = avg, fill = ethnicity, label =
  geom_col() +
  geom_text(vjust = 2,
            color = "white",
            size = 5) +
  scale_fill_hda() +
  scale_y_continuous(labels = label_dollar(scale = 0.001, suffix = "k"),
                     breaks = c(10000, 20000, 30000, 40000, 50000)) +
  add_zero_line("y") +
  theme_hda() +
  labs(
```

```
title = "Average amount of DPL assistance awarded by ethnicity",
subtitle = "Data from July 2017 through September 2022",
caption = "**Note:** Bars show total number of clients for that group."
)
```

## Average amount of DPL assistance awarded by Data from July 2017 through September 2022



**Note:** Bars show total number of clients for that group.

### Spatial patterns

#### ⚠ Note

Client and property location data is only available for clients who used the DPL program.

```
# Get list of all localities in Virginia
va_localities <- list_counties("VA")

# Get all census tracts for Virginia

va_tracts <- tracts(state = "VA") |>
  left_join(va_localities, by = c("COUNTYFP" = "county_code")) |>
```

```

st_transform(4502)

# Geocode client addresses

clients_geocode <- clients_all |>
  geocode(street = client_street_address, state = client_state,
          city = client_city, postalcode = client_zip,
          method = "geocodio",
          lat = lat,
          long = long) |>

# Drop SPARC entries with no address data

drop_na(lat) |>

# Specify WGS 1984 coordinate system for lat/long data

st_as_sf(coords = c("long", "lat"),
          remove = FALSE,
          crs = 4326) |>

# Project into Virginia State Plane South

st_transform(4502)

# Spatial join client data with tracts and localities

clients_sf <- st_join(clients_geocode, va_tracts) |>
  rename(Locality = county)

# Geocode property addresses

property_geocode <- clients_all |>
  geocode(street = property_street_address, state = property_state,
          city = property_city, postalcode = property_zip,
          method = "geocodio",
          lat = lat,
          long = long) |>

# Drop SPARC entries with no address data

```

```

drop_na(lat) |>

# Specify WGS 1984 coordinate system for lat/long data

st_as_sf(coords = c("long", "lat"),
         remove = FALSE,
         crs = 4326) |>

# Project into Virginia State Plane South

st_transform(4502)

# Spatial join client data with tracts and localities

property_sf <- st_join(property_geocode, va_tracts) |>
  rename(Locality = county)

# Save data

write_rds(clients_sf, "data/clients_sf.rds")
write_rds(property_sf, "data/property_sf.rds")

```

## Client origins

The map below shows the original addresses of each client, i.e. where they lived when they applied for DPL assistance.

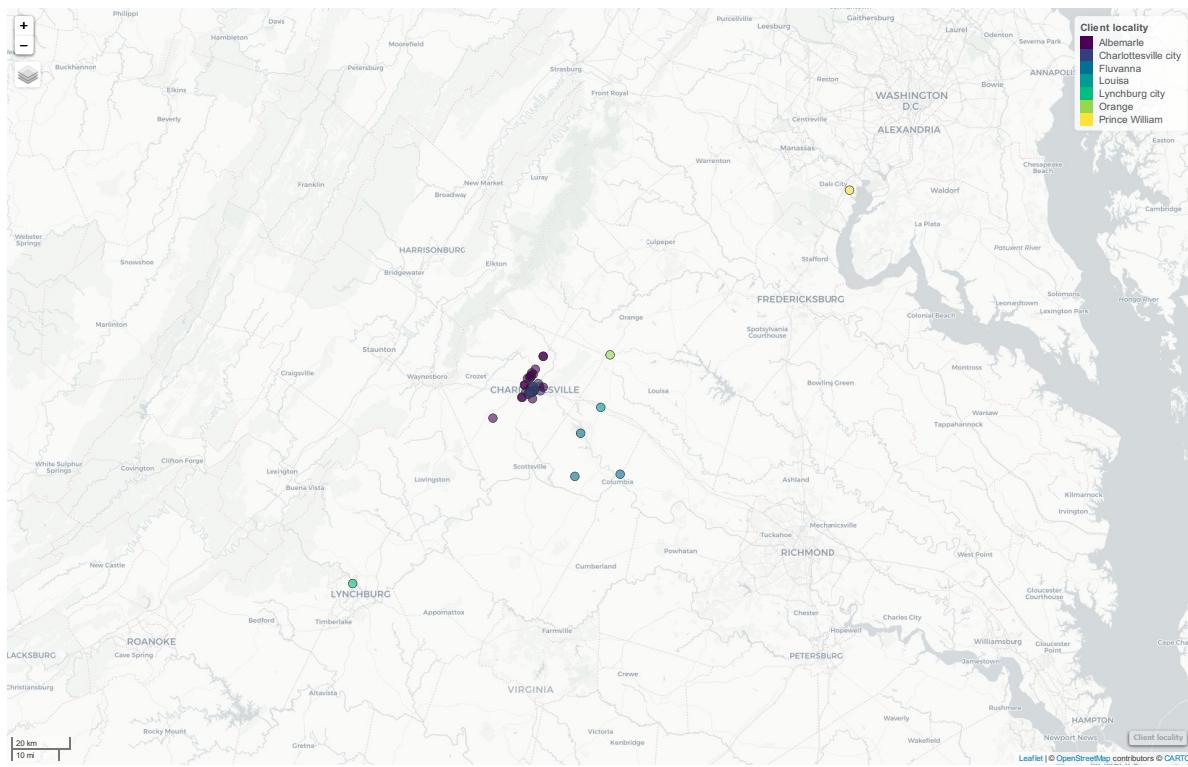
```

# Make map of clients

clients_sf <- read_rds("data/clients_sf.rds")

mapview(
  clients_sf,
  zcol = "Locality",
  layer.name = "Client locality",
  label = FALSE,
  popup = FALSE
)

```



Most clients lived in either Albemarle (20) or Charlottesville (17) when they applied to the DPL program. Three clients also came from Fluvanna, while all other origin localities has one client each.

```

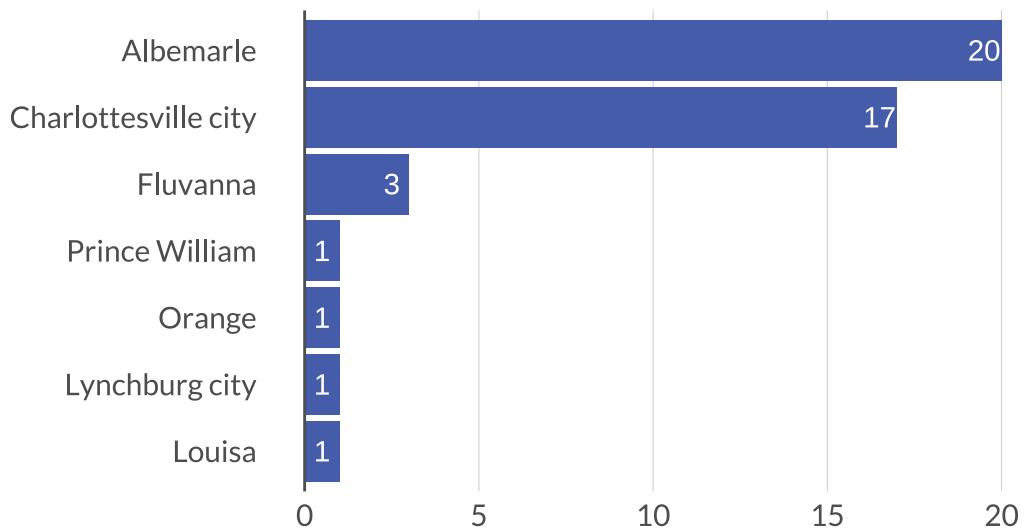
clients_locality <- clients_sf |>
  count(Locality)

ggplot(clients_locality, aes(x = n, y = reorder(Locality, n), label = n)) +
  geom_col(fill = "#445ca9") +
  geom_text(color = "#FFFFFF", nudge_x = -0.5) +
  theme_hda() +
  flip_gridlines() +
  add_zero_line("x") +
  labs(
    title = "Number of clients by locality of origin",
    subtitle = "Data from July 2017 through September 2022"
  )

```

## Number of clients by locality of origin

Data from July 2017 through September 2022



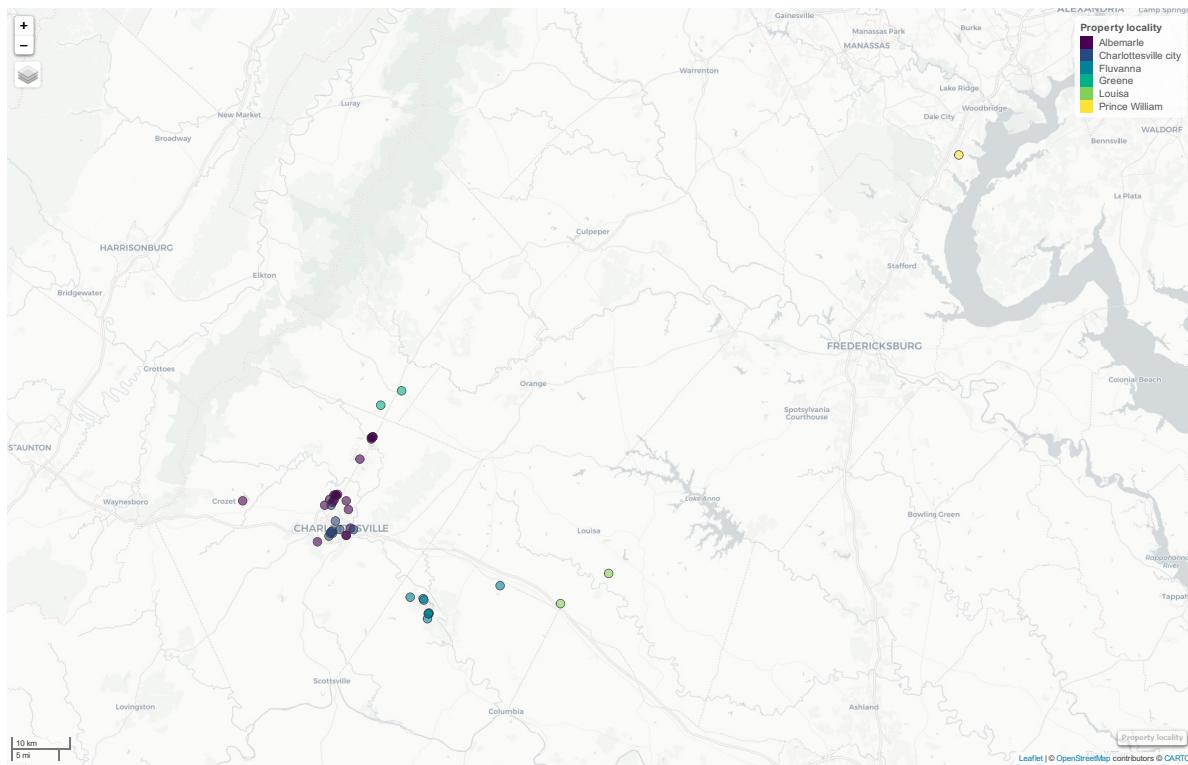
### Client destinations

The map below shows the addresses of homes purchased by clients, i.e. where they moved after using the DPL program.

```
# Make map of properties

property_sf <- read_rds("data/property_sf.rds")

mapview(
  property_sf,
  zcol = "Locality",
  layer.name = "Property locality",
  label = FALSE,
  popup = FALSE
)
```



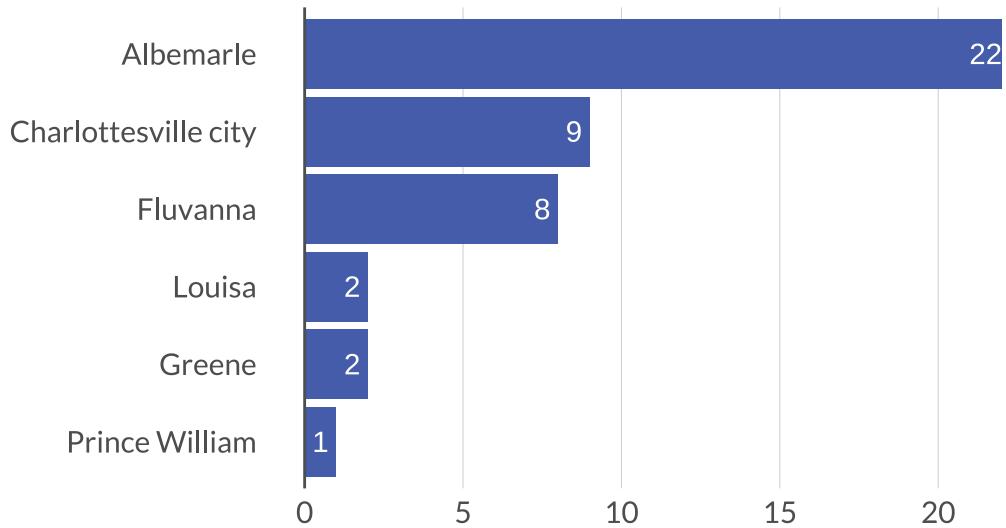
The majority of homes purchased by clients were in Albemarle (22), followed by Charlottesville (9) and Fluvanna (8). Two or fewer clients purchased homes in each of the other localities.

```
property_locality <- property_sf |>
  count(Locality)

ggplot(property_locality, aes(x = n, y = reorder(Locality, n), label = n)) +
  geom_col(fill = "#445ca9") +
  geom_text(color = "#FFFFFF", nudge_x = -0.5) +
  theme_hda() +
  flip_gridlines() +
  add_zero_line("x") +
  labs(
    title = "Number of clients by destination locality",
    subtitle = "Data from July 2017 through September 2022"
  )
```

# Number of clients by destination locality

Data from July 2017 through September 2022



## Client moves

Most of PHA's DPL clients moved from, moved to, or stayed in Albemarle County or the City of Charlottesville. A similar number of clients were originally from Albemarle (20) and Charlottesville (17), with only seven from elsewhere. Albemarle was the most common destination (22), followed by Charlottesville (9), and Fluvanna (8).

```
library(ggsankey)

client_moves <- st_drop_geometry(clients_sf) |>
  left_join(st_drop_geometry(property_sf), "id_phc") |>
  select("Origin" = 'Locality.x', "Destination" = 'Locality.y')

client_sankey <- client_moves |>
  make_long(Origin, Destination)

ggplot(client_sankey, aes(x = x, next_x = next_x,
                           node = node, next_node = next_node,
                           fill = node, label = node)) +
  geom_sankey(flow_alpha = 0.75, smooth = 15) +
  geom_sankey_label(fill = "white", colour = "#383c3d") +
```

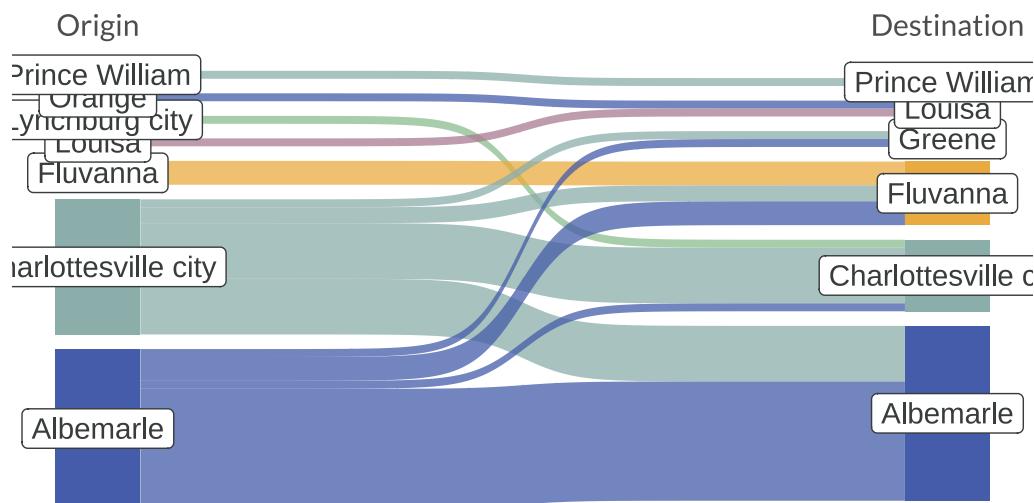
```

scale_fill_hda(repeat_pal = TRUE) +
  scale_x_discrete(expand = c(0.05, 0.05), position = "top") +
  theme_hda() +
  theme(axis.text.y = element_blank(),
        panel.grid.major.y = element_blank()) +
  labs(
    title = "Origin and destination of DPL clients",
    subtitle = "Data from July 2017 through September 2022"
  )

```

## Origin and destination of DPL clients

Data from July 2017 through September 2022



Among all clients from Albemarle, most stayed in the county, or to another county. Only one moved into Charlottesville. Just seven of the 17 clients from Charlottesville stayed in the city; the same number moved into Albemarle and the remainder to other counties.

```

client_moves_sum <- client_moves |>
  mutate(move = case_when(
    Origin == Destination ~ "Stayed",
    TRUE ~ "Moved"
  )) |>
  mutate(origin_sum = case_when(
    Origin %in% c("Fluvanna", "Louisa", "Lynchburg city", "Orange", "Prince William") ~ "F"
  ))

```

```

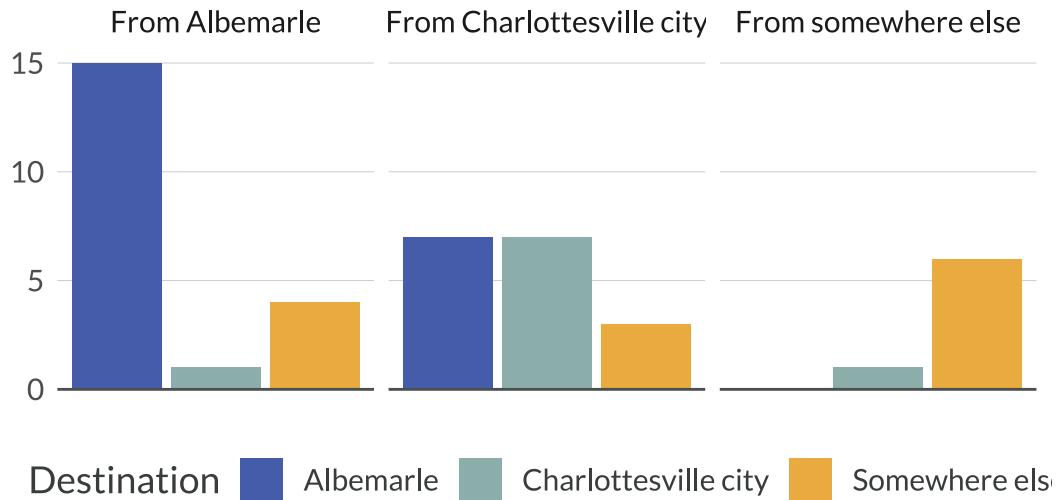
    TRUE ~ str_c("From ", Origin)
)) |>
mutate(destination_sum = case_when(
  Destination %in% c("Fluvanna", "Louisa", "Greene", "Prince William") ~ "Somewhere else"
  TRUE ~ Destination
))

ggplot(client_moves_sum, aes(x = destination_sum, fill = destination_sum)) +
  facet_wrap(~origin_sum) +
  geom_bar() +
  scale_fill_hda() +
  theme_hda() +
  add_zero_line("y") +
  theme(legend.position = "bottom",
        legend.title = element_text(),
        axis.text.x = element_blank()) +
  labs(
    title = "Number of clients by origin and destination",
    subtitle = "Data from July 2017 through September 2022",
    fill = "Destination"
)

```

## Number of clients by origin and destination

Data from July 2017 through September 2022



## Housing outcomes

The median sales price for all clients was \$224,500, but those prices vary significantly by program.

```
# Summarise home sales price by program

clients_inc_price <- clients_all |>

# Select only id_pha, program, income, home sales price, and race columns

select(1, 2, 9, 10, 20, 21, 53) |>

# Combine DLP and SPARC incomes into one column

mutate(income = case_when(
  !is.na(household_annual_income_dpl) & !is.na(household_annual_income_spc) ~ household_
  !is.na(household_annual_income_dpl) & is.na(household_annual_income_spc) ~ household_a
  is.na(household_annual_income_dpl) & !is.na(household_annual_income_spc) ~ household_a
  TRUE ~ NA_real_
)) |>

# Combine DLP and SPARC sales prices into one column

mutate(price = case_when(
  !is.na(sale_price_dpl) ~ sale_price_dpl,
  TRUE ~ sale_price_spc
))

# Generate table

clients_inc_price |>
  drop_na(price) |>
  group_by(program) |>
  summarise(n = n(),
            median = median(price),
            mean = mean(price)) |>
  mutate(across(3:4, ~ formattable::currency(.x, digits = 0))) |>
  arrange(desc(median)) |>
  kable(col.names = c("Program", "Clients", "Median sales price", "Mean sales price")) |>
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "hover", "condensed", "responsive")
```

Program	Clients	Median sales price	Mean sales price
Both	8	\$254,050	\$241,363
SPARC	86	\$228,800	\$226,923
DPL	36	\$184,753	\$200,497

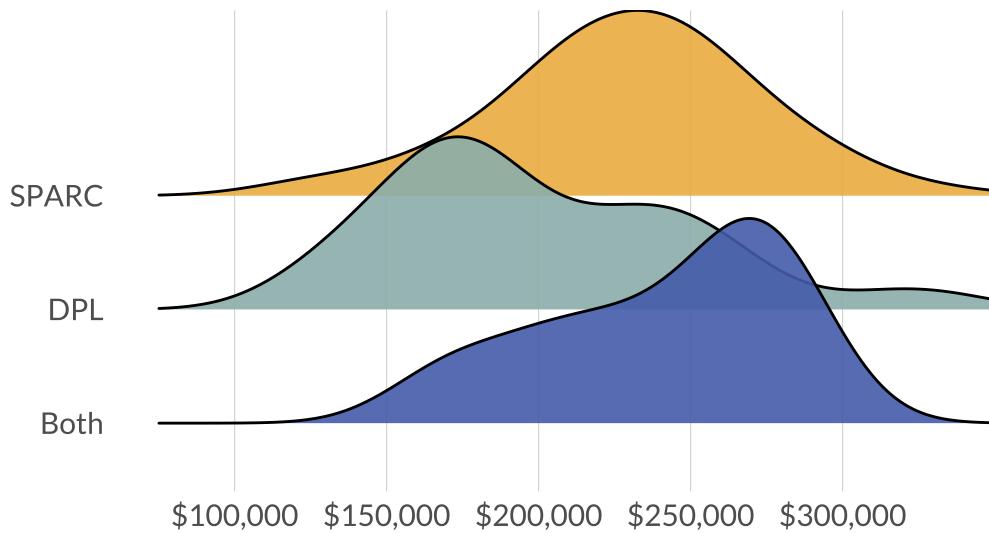
)

Home prices were lowest for DPL-only clients, with most between \$150,000 and \$200,000. SPARC-only clients purchased homes generally between \$200,000 and \$250,000. Clients who used both programs were able to purchase homes at higher prices, on average, than all other clients.

```
ggplot(clients_inc_price, aes(x = price, y = program, fill = program)) +
  geom_density_ridges(alpha = 0.9) +
  scale_fill_hda() +
  scale_x_continuous(labels = label_dollar(),
                     limits = c(75000, 350000),
                     breaks = c(100000, 150000, 200000, 250000, 300000)) +
  labs(title = "Client home purchase price by program",
       subtitle = "Data from July 2017 through September 2022") +
  theme_hda() +
  flip_gridlines()
```

## Client home purchase price by program

Data from July 2017 through September 2022



Average home prices also vary by race. Clients identifying as another race have the highest average, surpassing \$250,000, although this includes only three buyers. Clients whose records do not include their race also have high average prices around \$227,000. These are nearly all SPARC-only clients.

Asian and Black clients have average home prices slightly below the average for all clients, followed by clients who are White or who are White and Black. Those clients have average home prices well below \$200,000.

```
clients_price_race <- clients_inc_price |>
  drop_na(price) |>
  group_by(race) |>
  summarise(n = n(),
            Mean = mean(price),
            Median = median(price)) |>
  mutate(race = replace_na(race, "No data")) |>
  pivot_longer(3:4, names_to = "stat", values_to = "value")

ggplot(clients_price_race, aes(x = value, y = reorder(race, value),
                                fill = stat,
                                label = label_dollar(accuracy = 1, scale = 0.001, suffix =
geom_col(position = "dodge") +
```

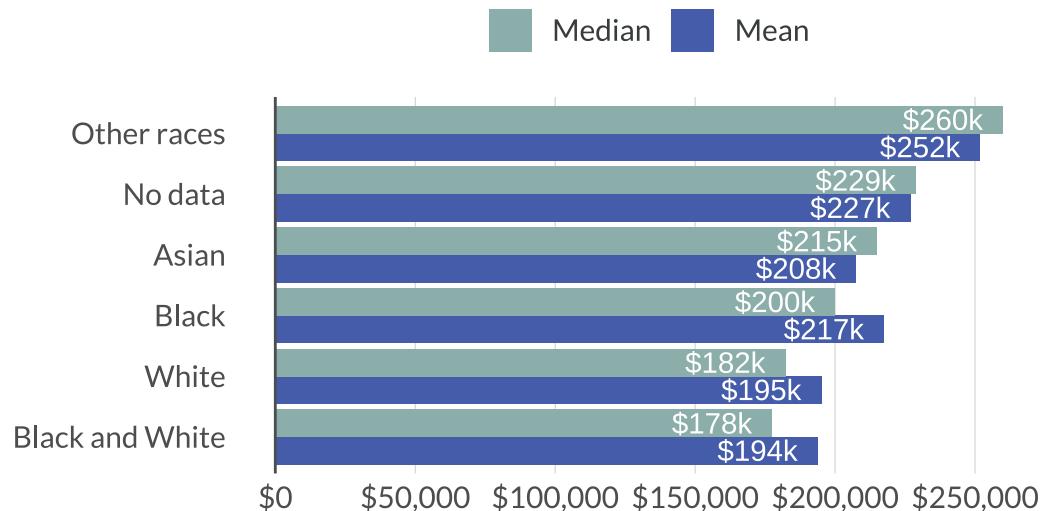
```

geom_text(color = "white", hjust = 1.25, position = position_dodge(0.9)) +
scale_fill_hda() +
scale_x_continuous(labels = label_dollar(),
                   breaks = c(0, 50000, 100000, 150000, 200000, 250000)) +
labs(title = "Average client home purchase price by race",
     subtitle = "Data from July 2017 through September 2022") +
theme_hda() +
flip_gridlines() +
add_zero_line("x") +
theme(legend.position = "top") +
guides(fill = guide_legend(reverse = TRUE))

```

## Average client home purchase price by race

Data from July 2017 through September 2022



```

# Client home attributes and price compared to neighborhood
# Client loan products compared to region (and neighborhood, if possible)

```

## **Home equity**

```
# Two methods to try and test

# Option 1:
# Find average percent change in assessment values from purchase date to current for all c
# Apply percent change to original sales price for estimated home value
# Calculate remaining mortgage balance using loan info
# value - balance = equity

# Option 2:
# Same as above but use MLS sales data
# Will have much smaller sample size -- may only be possible at city level
# Determine if any past clients have already sold home -- would provide most accurate esti
```

## **Takeaways**

Major (and other noteworthy) findings from analysis

## **Next steps**

HDAdvisors will continue to work with PHA on the following next steps:

- Develop consistent data entry/collection procedures to reduce time needed for data cleaning/correction in future analysis.
- Collect and analyze program application data for clients who did not receive assistance from DPL and/or SPARC. Assess, if possible, reasons for denial and compare with loan denial trends in HMDA data.

Plenty more to add here