My Organization's First R Package

1 — Working with Data in a Database

Why use databases?

Databases can be great. Here are some advantages:

– we know where the data is

– the data is in one place (in that, this is the source of truth)

– many people can have simultaneous access to the same data

– database tables can be updated with new data, with minimal disruption

– are there others? let me know!

Why use databases?

Databases can be *annoying* too. Here are some bad things I've experienced:

– queries can get stuck in the queue sometimes

– tables can be accidentally deleted or badly altered

– availability may only be through certain IP addresses

– some **dplyr** statements may not work at all (no SQL equivalent)

– others? (speak up those with bad experiences)

Which databases are available and (easily) accessible by R?

	MySQL MariaDB	Postgres Redshift	SQLite
R Package	`RMariaDB`	`RPostgres`	`RSQLite`
Website	`rmariadb.r-dbi.org`	`rpostgres.r-dbi.org`	`r-dbi.github.io/RSQLite`
Libraries Required			
Mac	`mysql-connector-c++`	`libpq`	—
Debian, Ubuntu	`libmysqlclient-dev` `Libmariadb-client-lgpl-dev`	`libpq-dev`	—
Fedora, CentOS, RHEL	`mysql-devel` `mariadb-devel`	`postgresql-devel`	—

Mac: `brew install <lib>` // Debian/Ubuntu: `sudo apt-get install -y <lib>` // Fedora et al.: `sudo yum install <lib>`

Which packages are required?

For our workshop, we'll need the `RMariaDB` package to access the **MySQL** database.

You'll also need to install some system libraries if you're using a **Mac** or **Linux** system.

Mac

```
brew install mysql-connector-c++
```

Debian/Ubuntu

```
sudo apt-get install -y libmysqlclient-dev
sudo apt-get install -y Libmariadb-client-lgpl-dev
```

Fedora et al.

```
sudo yum install mysql-devel
sudo yum install mariadb-devel
```

Which packages are required?

We need the **DBI** package. It can be thought of as the central package for connecting to databases in **R**.

We also need packages from the *Tidyverse*. In particular, **dplyr** and **dbplyr** are essential for working with databases.

R Package	`DBI`	`dplyr`	`dbplyr`
Website	`dbi.r-dbi.org`	`dplyr.tidyverse.org`	`dbplyr.tidyverse.org`

CODE DEMO: `01_01_databases.Rmd`

We are going to move over to the `.Rmd` file named:

`exercises/01_01_databases.Rmd`

In that, we will create a simple **SQLite** database, create a table with `nycflights13::flights` data, and use **dplyr** to work with the in-memory DB table.

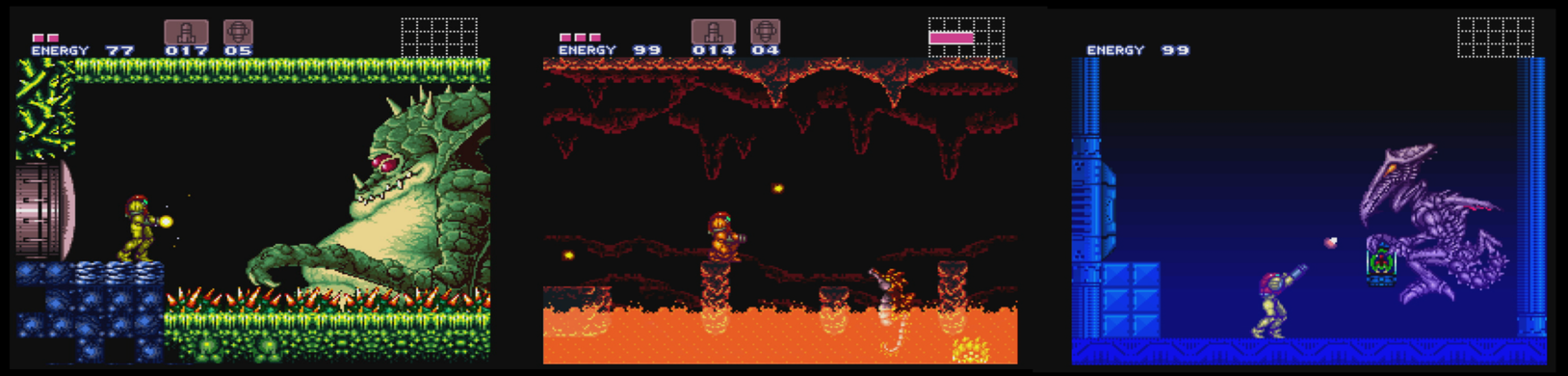
Introducing our database and our data tables.

Just for today, we all work for a company called *Intendo*.

INTENDO

FINE MOBILE GAMES

They have a database with data available for the **2015** year.
It has all sorts of information for their only game: *Super Jetroid*.

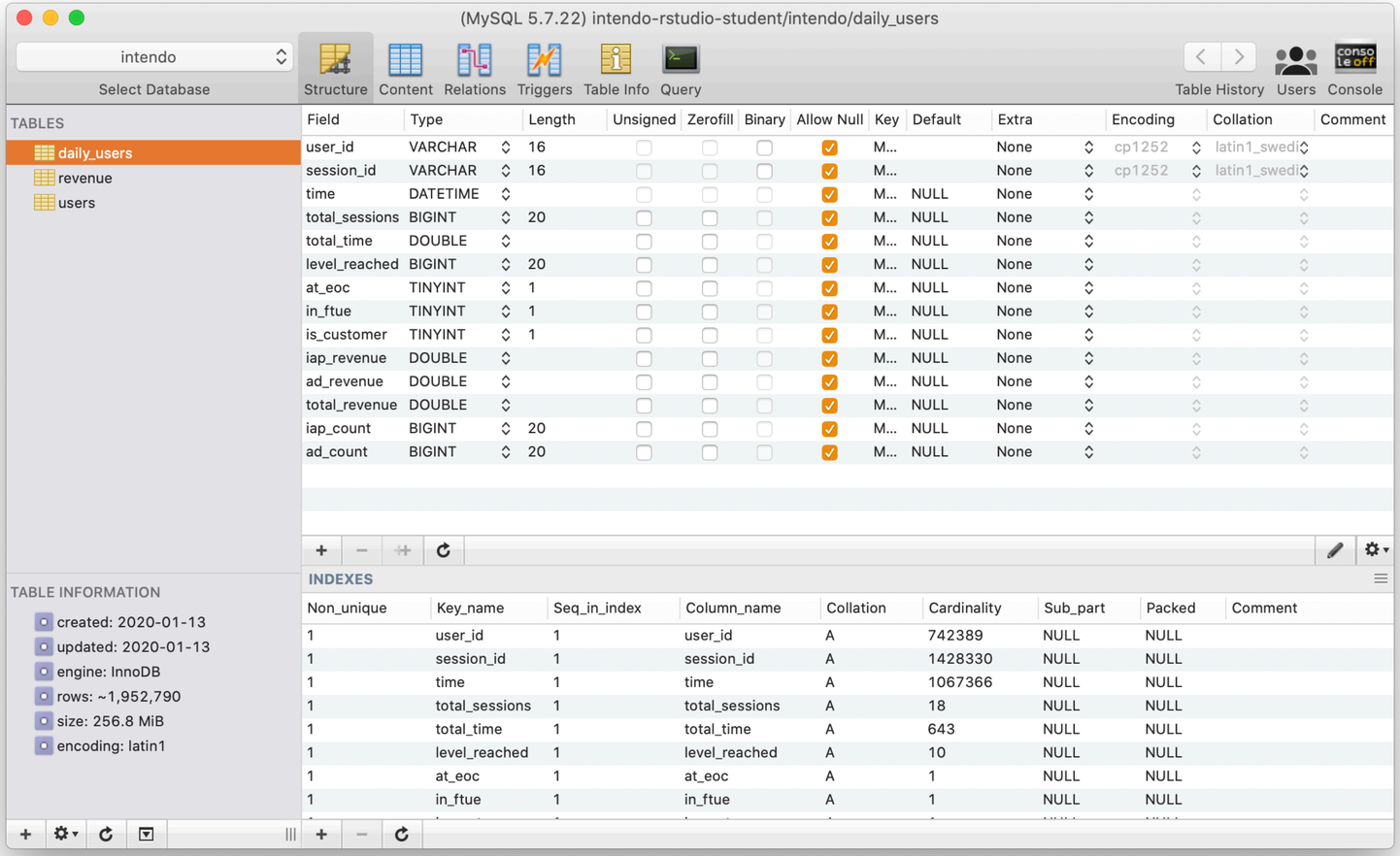


Introducing our database and our data tables.

The database is hosted on AWS using RDS (Relational Database Service).

It's a **MySQL** database with 3 tables containing all of their user data and revenue data.

This database is publicly accessible. In most organizations, however, access is only possible when at specific IP addresses.



Introducing our database and our data tables.

There are three tables: `daily_users`, `revenue`, and `users`.

`daily_users`

— updated daily, has a record per user, per session (could be multiple logins per day)
— each records contains info on user's past activity and revenue totals

TABLE COLUMNS

`user_id`
`session_id`
`time`

IDs for the user and for the session; login date-time.

`total_sessions`
`total_time`

summary totals on number of sessions and total time active (in minutes).

`level_reached`
`at_eoc`
`in_ftue`

Progression info: What level? At the end of content? In the first-time user experience?

`is_customer`
`iap_revenue`
`ad_revenue`
`total_revenue`

Has user spent *any* money up to now? USD amounts for IAPs, ad views, and sum of both.

`iap_count`
`ad_count`

Count of IAPs bought, count of ads viewed.

Introducing our database and our data tables.

There are three tables: `daily_users`, `revenue`, and `users`.

`revenue`

— updated daily, has a record IAP bought or ad viewed by a user
— contains specifics on the name of IAP, ad type, and price for IAP, and actual revenue

TABLE COLUMNS

`user_id`
`session_id`
`time`

IDs for the user and for the session; login date-time.

`name`
`size`

Name of the IAP or ad type. The `size` is relative worth of the IAP.

`type`

The type of IAP or `"ad"` for advertising revenue. IAP types are: `"currency"`, `"season_pass"`, or `"offer_agent"`.

`price`

The price in USD for the IAP (`NULL` if type is `"ad"`).

`revenue`

The actual revenue received for the ad view or IAP purchase.

Introducing our database and our data tables.

There are three tables: `daily_users`, `revenue`, and `users`.

`users`

— updated daily, a user summary table with one row per user
— contains specifics on user first login, revenue earned, primary device, and country

TABLE COLUMNS

`user_id`
`first_login`

ID for the user and first login date-time.

`iap_count`
`iap_revenue`
`ad_count`
`ad_revenue`
`total_revenue`

Revenue from and counts of IAPs and ad views.

`customer`
`subscriber`
`first_iap`

Has user ever bought an IAP? Ever bought a `"season_pass"`? What was the first IAP?

`platform`
`device`

What was the login device and its platform?

`acquired`
`country`

How was the user acquired, through a campaign or `"organic"`? What country is the user from?

CODE DEMO: `01_02_databases.Rmd`

We are going to move over to the `.Rmd` file named:

`exercises/01_02_databases.Rmd`

We'll learn how to connect to the *Intendo*'s `intendo` database.
It involves making a *connection* object.

We'll also learn about how to list the database tables.
That involves using the `DBI::dbListTables()` function.

YOUR TURN (WORKING ON THE intendo PACKAGE)

30 minutes

Open up the package-in-progress at:

`p_01_intendo/01_intendo.Rproj`

Take a look around the package and see which components are already in place.

There are no functions yet (No `R` folder!) but there is a `scripts` folder with `.R` files that contain functions. Test them out and then make them part of the package.

Use some of these **usethis** functions:

ESSENTIAL `use_r()` `use_pipe()` `use_build_ignore()` `use_package()`

BONUS `use_package_doc()` `use_news_md()` `use_readme_rmd()` `use_roxygen_md()`

Discuss things with your neighbors, talk to the instructors and TAs… we'll help!