(1)The phase I writeup is attached to the end of this document.

(2) Team members: Chris Wang (315), Nian Gao (315)
Email: lwang178@jhu.edu, ngao6@jhu.edu
Description: This astronomical database systematically organizes planetary and stellar data, storing fundamental properties of planets alongside their associated stellar systems and molecular characteristics.
All the codes, database file, and other files related to this project is stored at:
https://github.com/Chrrrrris/StarPlanet_Abundance_Catalog

(3) We have made multiple changes since the phase I submission, including removing unnecessary attributes and changing the database structures to include more tables to represent more relations in the database and reduce data duplication in the original flat data (thus some queries written in phase I are no longer applicable). We also switched our focus from backend integration to application of advanced SQL tools (triggers) and complex data extraction from online resources.

(4)
We downloaded csv files from online resources. The code we wrote to preprocess the data is located in 'code -> data_preprocessing.ipynb.'

1. The planet composition data (location: 'original data -> iac_exoplanet_atmospheres-20241215.csv' comes from: https://research.iac.es/proyecto/exoatmospheres/table.php
2. Relation between planet and star systems & other data related to planets and system (location: 'original data -> PS_2024.12.13_18.57.54.csv' comes from: https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=PS&constraint=disc_facility+like+%27%25TESS%25%27
3. Elemental Abundances for stars in Kepler Objects of Interests (KOI): extracted from Brewer et al. 2018
   'Original data -> brewer_abundance.vot'
   https://ui.adsabs.harvard.edu/abs/2018ApJS..237...38B/abstract
4. Elemental Abundances from APOGEE (Apache Point Observatory Galactic Evolution Experiment). We performed quality flag queries on SDSS SkyServer https://skyserver.sdss.org/dr18/SearchTools/sql
   The queries performed can be seen at 'code -> sdss.sql'. The data output can be seen at 'original data->APOGEE.csv'
5. A list of all stars' IDs is queried through the SIMBAD catalog via a python interface, which enables accurate query operations. https://simbad.u-strasbg.fr/simbad/sim-fbasic

(5) We used SQLite to create and manage the database.

(6) All the codes are in the ‘code’ folder and can run easily. To access our database, users can simply run *sqlite3 StarPlanet.db* to perform simple query operations.

(7) We explored advanced SQL topics, specifically triggers, in our database. We implemented a number of triggers to: a) ensure values inserted are in the expected format; b) ensure values inserted are valid; c) perform cascade delete when the deleted record affects other tables.
We performed complex extraction of real data from online sources. Utilizing our observational astronomy knowledge and by referring to Reeves et al. 2023, we performed a carefully curated stellar elemental abundance data extraction from the SDSS server to ensure our elemental abundances are robust and free from observational/data analysis artifacts. The methods we applied can be seen at ‘code->sdss.sql’. Moreover, because the IDs for astronomical objects are in most cases not one-to-one, we performed interactions with the SIMBAD catalog to make sure we covered all the stars that are observed by different surveys that put them under different IDs. Details of how we do that can be seen at ‘code->load_database.ipynb’

(8)

1. Our database robustly connects different astronomical surveys that output data under different schemas. By implementing codes that deal with those heterogeneities, we provide an automatic pipeline that can update the database in a timely manner when new generations of those survey data are released.
2. Astronomical objects have different IDs. By querying the SIMBAD catalog with object’s ID from exoplanet archives, we get a list of all IDs of those astronomical objects, which are stored in the host_ids table. In this way, we ensure all the query operations within our database are accurate across a variety of astronomical surveys.
3. Our database is the first database that attempts to connect the composition of planets and their hosting stars. By comparing the compositions of these two entities, we can derive valuable knowledge about how the planets could be formed (e.g. Lothringer et al. 2021). This is an extremely novel and new area of study in the field of exoplanets. With more exoplanets scheduled to be characterized in the next few years with JWST and HST, the amount of data in this area will explode and we will be able to explore a more complete parameter space. By providing a database that logs the elemental abundances of planets and their host stars, our project is one of the first steps towards advancing theories of planet formation with comparative stellar-exoplanetology.
4. Even though our database connecting stars’ and planets’ abundance is still minimal (due to the small number of planets that are characterized), we have constructed a database that logs the elemental abundances and stellar properties of all exoplanet-hosting stars surveyed by both Brewer et al. and APOGEE as a byproduct of our effort. This is also the first database that collects the properties of exoplanet-hosting stars. This database can also help answer a range of scientific questions. For example, one can look at the physical properties of exoplanets and the elemental abundances of their hosting stars and examine whether these two are correlated in any dimension. As an example, Chris did an independent project examining whether planet formation is correlated with stellar elemental abundances (Wang & Schlaufman in prep). If there had been a database like

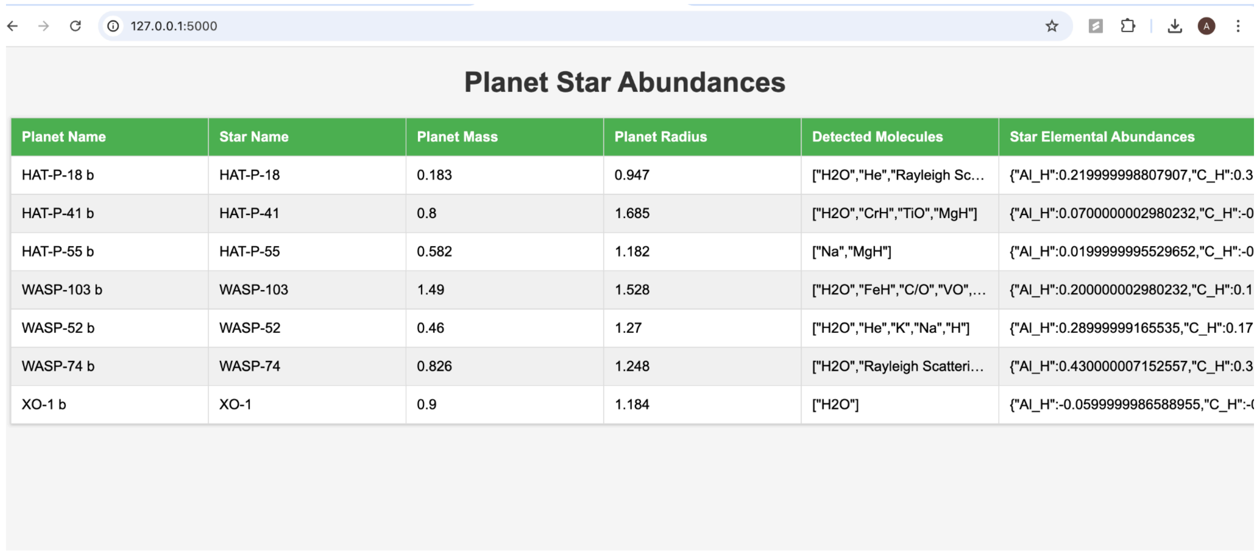
this that existed while Chris was doing that project, the progress of this project could have been largely expedited.

(9)

1. For planets' atmospheres, we didn't provide the abundance of the species but only provided which species have been constrained. There has not been a database that catalogs the abundance of atmospheric species of exoplanets, as the formats of those abundances are not standardized across literature and therefore a fully automatic extraction is not really possible. In the future, we will try to sample and standardize these results manually
2. For the user interface, we only show the table that links planet and stars' abundance. As a future work, we allow user to specify which table to show, whether it's stellar properties and abundance from Brewer or those from APOGEE. We would also enable simple sorting and searching based on column values.
3. Our database is intentionally made small to enhance its performance due to our limited computing power. In the future, we will integrate the full exoplanet archive and potentially APOGEE stellar catalogs to enable complete coverage of all the exoplanets and all the stars with elemental abundance measurements. We will be able to do that when we have access to our research group's server.

(10) N/A

(11)
The frontend:



127.0.0.1:5000

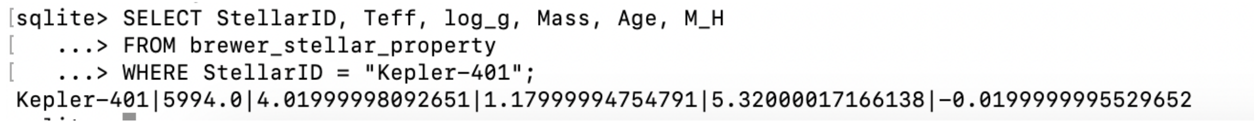
Planet Star Abundances

Planet Name	Star Name	Planet Mass	Planet Radius	Detected Molecules	Star Elemental Abundances
HAT-P-18 b	HAT-P-18	0.183	0.947	["H2O","He","Rayleigh Sc…	{"Al_H":0.219999998807907,"C_H":0.3
HAT-P-41 b	HAT-P-41	0.8	1.685	["H2O","CrH","TiO","MgH"]	{"Al_H":0.0700000002980232,"C_H":-0
HAT-P-55 b	HAT-P-55	0.582	1.182	["Na","MgH"]	{"Al_H":0.0199999995529652,"C_H":-0
WASP-103 b	WASP-103	1.49	1.528	["H2O","FeH","C/O","VO",…	{"Al_H":0.200000002980232,"C_H":0.1
WASP-52 b	WASP-52	0.46	1.27	["H2O","He","K","Na","H"]	{"Al_H":0.28999999165535,"C_H":0.17
WASP-74 b	WASP-74	0.826	1.248	["H2O","Rayleigh Scatteri…	{"Al_H":0.430000007152557,"C_H":0.3
XO-1 b	XO-1	0.9	1.184	["H2O"]	{"Al_H":-0.0599999986588955,"C_H":-

This frontend interface displays a selected view of the database. Users can hover over the cells to see the truncated values. This view is constructed based on created entities in our database, combining stellar elemental abundances and detected species of the planets.

We also built a table that displays exoplanet hosting stars' properties and elemental abundances from both Brewer and APOGEE.
As an example, let's say we call

SELECT Teff, log_g, Mass, Age, M_H FROM brewer_stellar_property WHERE StellarID = “Kepler-401”;

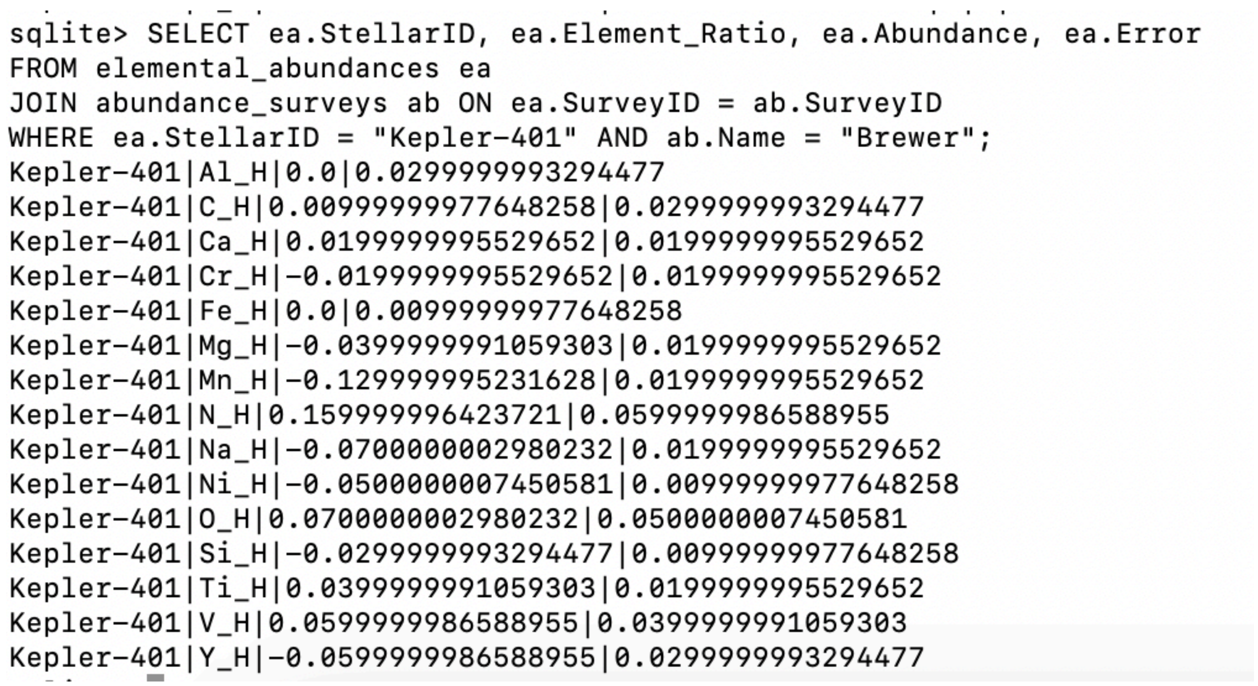


```
sqlite> SELECT StellarID, Teff, log_g, Mass, Age, M_H
   ...> FROM brewer_stellar_property
   ...> WHERE StellarID = "Kepler-401";
Kepler-401|5994.0|4.01999998092651|1.17999994754791|5.32000017166138|-0.0199999995529652
```

We derive the effective temperature, log g, mass, age, and metallicity for Kepler-401, an exoplanet hosting star that we have confirmed from the Exoplanet Archive, from the survey from Brewer et al. 2018.

Let’s say we also want elemental abundances for this star from Brewer. We can call

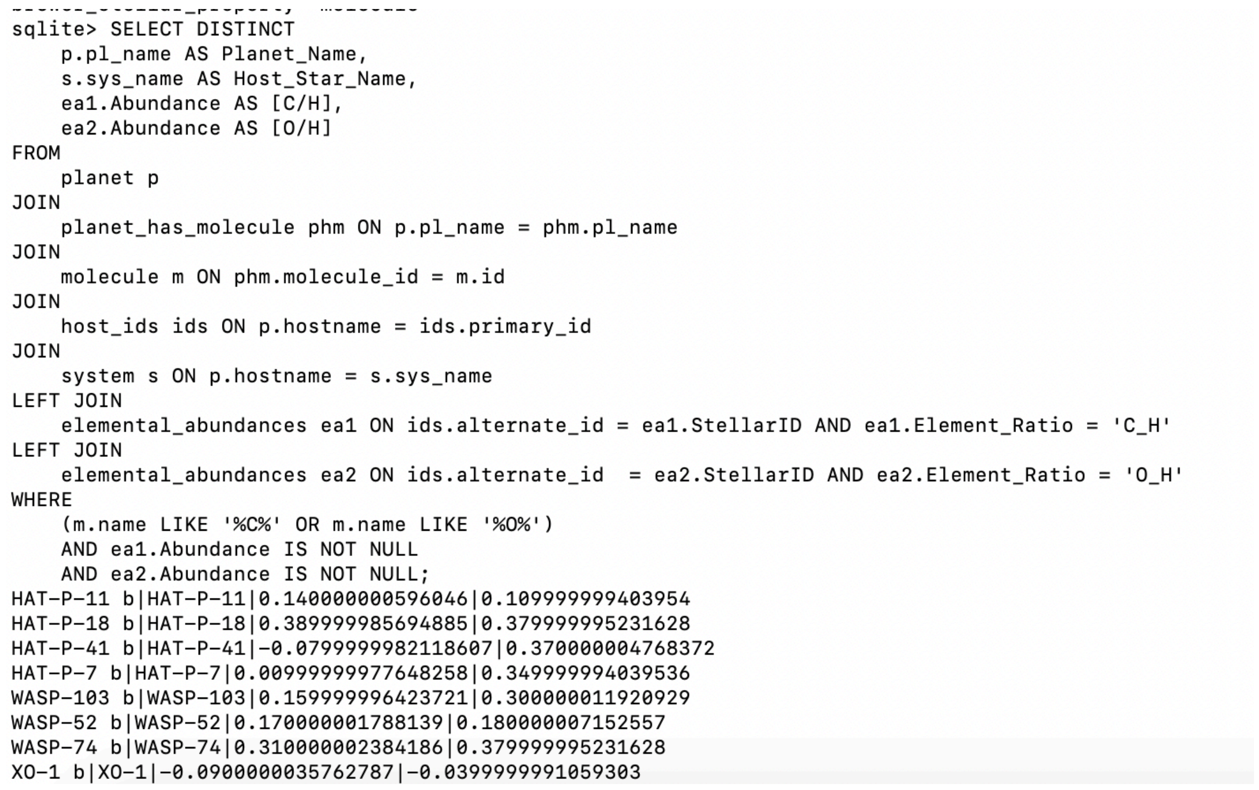
SELECT ea.StellarID, ea.Element_Ratio, ea.Abundance, ea.Error
FROM elemental_abundances ea
JOIN abundance_surveys ab ON ea.SurveyID = ab.SurveyID
WHERE ea.StellarID = "Kepler-401" AND ab.Name = "Brewer";



```
sqlite> SELECT ea.StellarID, ea.Element_Ratio, ea.Abundance, ea.Error
FROM elemental_abundances ea
JOIN abundance_surveys ab ON ea.SurveyID = ab.SurveyID
WHERE ea.StellarID = "Kepler-401" AND ab.Name = "Brewer";
Kepler-401|Al_H|0.0|0.0299999993294477
Kepler-401|C_H|0.00999999977648258|0.0299999993294477
Kepler-401|Ca_H|0.0199999995529652|0.0199999995529652
Kepler-401|Cr_H|-0.0199999995529652|0.0199999995529652
Kepler-401|Fe_H|0.0|0.00999999977648258
Kepler-401|Mg_H|-0.0399999991059303|0.0199999995529652
Kepler-401|Mn_H|-0.129999995231628|0.0199999995529652
Kepler-401|N_H|0.159999996423721|0.0599999986588955
Kepler-401|Na_H|-0.0700000002980232|0.0199999995529652
Kepler-401|Ni_H|-0.0500000007450581|0.00999999977648258
Kepler-401|O_H|0.0700000002980232|0.0500000007450581
Kepler-401|Si_H|-0.0299999993294477|0.00999999977648258
Kepler-401|Ti_H|0.0399999991059303|0.0199999995529652
Kepler-401|V_H|0.0599999986588955|0.0399999991059303
Kepler-401|Y_H|-0.0599999986588955|0.0299999993294477
```

If a researcher is only interested in finding an exoplanet hosting star’s property, our database creates a quick way to look up those values. If they want to study planet formation pathways by comparing star’s C/O and the planet’s C/O, one can directly look at the frontend table or perform the following query:

```
SELECT DISTINCT
    p.pl_name AS Planet_Name,
    s.sys_name AS Host_Star_Name,
   ea1.Abundance AS [C/H],
   ea2.Abundance AS [O/H]
FROM
   planet p
JOIN
   planet_has_molecule phm ON p.pl_name = phm.pl_name
JOIN
   molecule m ON phm.molecule_id = m.id
JOIN
   host_ids ids ON p.hostname = ids.primary_id
JOIN
   system s ON p.hostname = s.sys_name
LEFT JOIN
   elemental_abundances ea1 ON ids.alternate_id = ea1.StellarID AND ea1.Element_Ratio =
'C_H'
LEFT JOIN
   elemental_abundances ea2 ON ids.alternate_id  = ea2.StellarID AND ea2.Element_Ratio =
'O_H'
WHERE
   (m.name LIKE '%C%' OR m.name LIKE '%O%')
   AND ea1.Abundance IS NOT NULL
   AND ea2.Abundance IS NOT NULL;
```

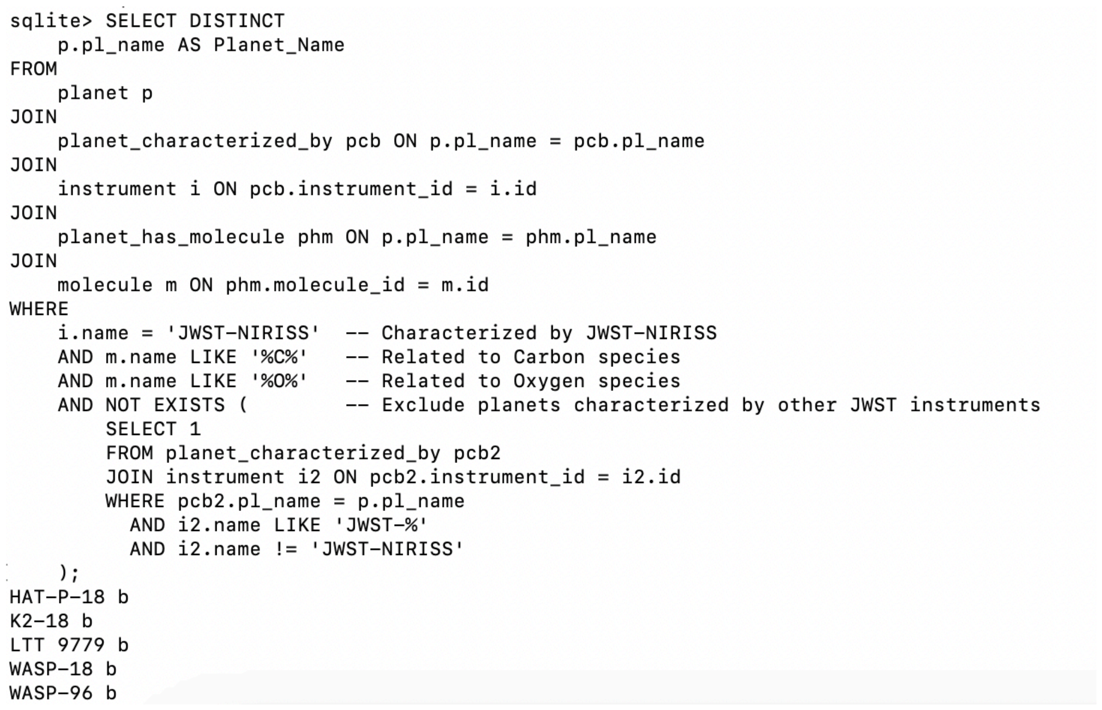


```
sqlite> SELECT DISTINCT
    p.pl_name AS Planet_Name,
    s.sys_name AS Host_Star_Name,
    ea1.Abundance AS [C/H],
    ea2.Abundance AS [O/H]
FROM
    planet p
JOIN
    planet_has_molecule phm ON p.pl_name = phm.pl_name
JOIN
    molecule m ON phm.molecule_id = m.id
JOIN
    host_ids ids ON p.hostname = ids.primary_id
JOIN
    system s ON p.hostname = s.sys_name
LEFT JOIN
    elemental_abundances ea1 ON ids.alternate_id = ea1.StellarID AND ea1.Element_Ratio = 'C_H'
LEFT JOIN
    elemental_abundances ea2 ON ids.alternate_id  = ea2.StellarID AND ea2.Element_Ratio = 'O_H'
WHERE
    (m.name LIKE '%C%' OR m.name LIKE '%O%')
    AND ea1.Abundance IS NOT NULL
    AND ea2.Abundance IS NOT NULL;
HAT-P-11 b|HAT-P-11|0.140000000596046|0.109999999403954
HAT-P-18 b|HAT-P-18|0.389999985694885|0.379999995231628
HAT-P-41 b|HAT-P-41|-0.0799999982118607|0.370000004768372
HAT-P-7 b|HAT-P-7|0.00999999977648258|0.349999994039536
WASP-103 b|WASP-103|0.159999996423721|0.300000011920929
WASP-52 b|WASP-52|0.170000001788139|0.180000007152557
WASP-74 b|WASP-74|0.310000002384186|0.379999995231628
XO-1 b|XO-1|-0.0900000035762787|-0.0399999991059303
```

Note that we crossmatched different surveys (exoplanet archive and elemental abundances from brewer) by matching all the star's ids logged at host_ids, to make sure the queried results are complete.

We can quickly examine what planets have been characterized C/O with JWST-NIRISS but not with other JWST instruments. Because the C/O from JWST-NIRISS is biased and incomplete due to the limited wavelength coverage, future work is needed to better characterize C/O with other JWST instruments. Thus this table is valuable from an observational perspective

```
SELECT DISTINCT
  p.pl_name AS Planet_Name
FROM
  planet p
JOIN
  planet_characterized_by pcb ON p.pl_name = pcb.pl_name
JOIN
  instrument i ON pcb.instrument_id = i.id
JOIN
  planet_has_molecule phm ON p.pl_name = phm.pl_name
JOIN
  molecule m ON phm.molecule_id = m.id
WHERE
  i.name = 'JWST-NIRISS'  -- Characterized by JWST-NIRISS
  AND m.name LIKE '%C%'   -- Related to Carbon species
  AND m.name LIKE '%O%'   -- Related to Oxygen species
  AND NOT EXISTS (        -- Exclude planets characterized by other JWST instruments
    SELECT 1
    FROM planet_characterized_by pcb2
    JOIN instrument i2 ON pcb2.instrument_id = i2.id
    WHERE pcb2.pl_name = p.pl_name
     AND i2.name LIKE 'JWST-%'
     AND i2.name != 'JWST-NIRISS'
  );
```



```
sqlite> SELECT DISTINCT
    p.pl_name AS Planet_Name
FROM
    planet p
JOIN
    planet_characterized_by pcb ON p.pl_name = pcb.pl_name
JOIN
    instrument i ON pcb.instrument_id = i.id
JOIN
    planet_has_molecule phm ON p.pl_name = phm.pl_name
JOIN
    molecule m ON phm.molecule_id = m.id
WHERE
    i.name = 'JWST-NIRISS'  -- Characterized by JWST-NIRISS
    AND m.name LIKE '%C%'   -- Related to Carbon species
    AND m.name LIKE '%O%'   -- Related to Oxygen species
    AND NOT EXISTS (        -- Exclude planets characterized by other JWST instruments
        SELECT 1
        FROM planet_characterized_by pcb2
        JOIN instrument i2 ON pcb2.instrument_id = i2.id
        WHERE pcb2.pl_name = p.pl_name
          AND i2.name LIKE 'JWST-%'
          AND i2.name != 'JWST-NIRISS'
    );
HAT-P-18 b
K2-18 b
LTT 9779 b
WASP-18 b
WASP-96 b
```

(12)

```
CREATE TABLE IF NOT EXISTS system (
        sys_name VARCHAR(100) PRIMARY KEY,
        sy_snum INTEGER,
        sy_pnum INTEGER,
        ra FLOAT,
        dec FLOAT
    )
# INSERT INTO system VALUES ('51 Eri', 3, 1, 69.4007424, -2.4738245);

CREATE TABLE IF NOT EXISTS discovery (
        disc_refname VARCHAR(200) PRIMARY KEY,
        disc_pubdate DATE
    )
# INSERT INTO discovery VALUES (‘<a refstr=GAIDOS_ET_AL__2021
href=https://ui.adsabs.harvard.edu/abs/2021MNRAS.tmp.2819G/abstract target=ref>Gaidos et
al. 2021</a>’, Oct-21);

CREATE TABLE IF NOT EXISTS molecule (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        name VARCHAR(100) UNIQUE
    )
# INSERT INTO molecule (name) VALUES ('H2O');
```

```
CREATE TABLE IF NOT EXISTS planet (
        pl_name VARCHAR(100) PRIMARY KEY,
        hostname VARCHAR(100),
        mass FLOAT,
        radius FLOAT,
        orbital_period FLOAT,
        tsm FLOAT,
        disc VARCHAR(200),
        FOREIGN KEY (hostname) REFERENCES system(sys_name),
        FOREIGN KEY (disc) REFERENCES discovery(disc_refname)
    )
# INSERT INTO planet VALUES (‘2M0437 b’, ‘2MASS J04372171+2651014’, 4.0, 10260.0,
1668.9209, ‘<a refstr=GAIDOS_ET_AL__2021
href=https://ui.adsabs.harvard.edu/abs/2021MNRAS.tmp.2819G/abstract target=ref>Gaidos et
al. 2021</a>’);

CREATE TABLE IF NOT EXISTS planet_has_molecule (
        pl_name VARCHAR(100),
        molecule_id INTEGER,
        FOREIGN KEY (pl_name) REFERENCES planet(pl_name),
        FOREIGN KEY (molecule_id) REFERENCES molecule(id),
        PRIMARY KEY (pl_name, molecule_id)
    )
# INSERT INTO planet_has_molecule VALUES (‘2M0437 b’, 1);

CREATE TABLE IF NOT EXISTS instrument (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        name VARCHAR(100) UNIQUE
    )
# INSERT INTO instrument(name) VALUES (‘JWST-NIRISS’);

CREATE TABLE IF NOT EXISTS planet_characterized_by (
        pl_name VARCHAR(100),
        instrument_id INTEGER,
        FOREIGN KEY (pl_name) REFERENCES planet(pl_name),
        FOREIGN KEY (instrument_id) REFERENCES instrument(id),
        PRIMARY KEY (pl_name, instrument_id)
    )
# INSERT INTO planet_characterized_by VALUES (‘2M 0103-55 (AB) b’, 14);

CREATE TABLE IF NOT EXISTS host_ids (
        primary_id VARCHAR(100),
        alternate_id VARCHAR(100),
        FOREIGN KEY (primary_id) REFERENCES system(sys_name)
        PRIMARY KEY (primary_id, alternate_id)
    )
# INSERT INTO host_ids VALUES (‘TOI-270’, ‘CNS5 1134’);

CREATE TABLE IF NOT EXISTS brewer_stellar_property (
        StellarID TEXT PRIMARY KEY,
        Teff FLOAT,
        log_g FLOAT,
        SN FLOAT,
        Mass FLOAT,
        l_Mass FLOAT,
        u_Mass FLOAT,
        Age FLOAT,
        l_Age FLOAT,
        u_Age FLOAT,
        M_H FLOAT,
        e_M_H FLOAT
    )
# INSERT INTO brewer_stellar_property VALUES (‘KOI-3248’, 5742.0, 4.34000015258789,
50.0, 0.959999978542328, 0.949999988079071, 0.97000002861023, 8.39999961853027,
8.01000022888184, 8.72999954223633, -0.0700000002980232, 0.00999999977648258);

CREATE TABLE IF NOT EXISTS apogee_stellar_property (
        StellarID TEXT PRIMARY KEY,
        Teff FLOAT,
        e_Teff FLOAT,
        log_g FLOAT,
        e_log_g FLOAT,
        SN FLOAT,
        M_H FLOAT,
        e_M_H FLOAT
    )
# INSERT INTO apogee_stellar_property VALUES (‘2M16100392+2644336’, 4805.314,
9.910008, 2.627599, 0.02574208, 162.0193, -0.16702, 0.00714802);

CREATE TABLE IF NOT EXISTS elemental_abundances (
        StellarID VARCHAR(100) NOT NULL,
        Element_Ratio VARCHAR(20) NOT NULL,
        Abundance FLOAT,
        Error FLOAT,
        SurveyID INTEGER,
        FOREIGN KEY (StellarID) REFERENCES brewer_stellar_property(StellarID),
        FOREIGN KEY (SurveyID) REFERENCES abundance_surveys(SurveyID),
        UNIQUE(StellarID, Element_Ratio)
    )
# INSERT INTO elemental_abundances VALUES (‘KOI-3248’, ‘C_H’, -0.0599999986588955,
0.0299999993294477, 1);
```

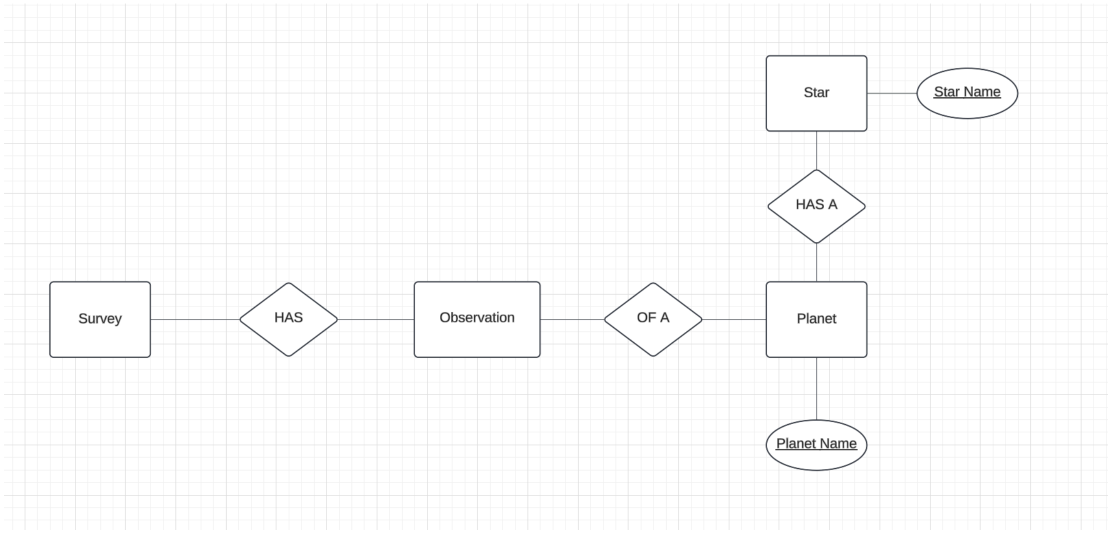
```
CREATE TABLE IF NOT EXISTS abundance_surveys (
        SurveyID INTEGER PRIMARY KEY AUTOINCREMENT,
        Name VARCHAR(100) UNIQUE
    )
# INSERT INTO abundance_surveys(Name) VALUES (‘Brewer’);
```

(13) The SQL code used to load the database is in load_database.ipynb. Other SQL codes are all in the ‘code’ folder.

Project Phase I

1) Chris Wang, Nian Gao
2) An astronomy database
3) List of queries:
    a) List all the currently confirmed terrestrial planets (Rp < 1.7 Me)
    b) List all the planets around M-class stars
    c) List all the planets observed by both transits and radial velocity methods
    d) List all the giant planets' names and the metallicities of their hosting stars from APOGEE
    e) List the TSM of all the planets characterized by JWST
    f) List all the planets not observed by JWST but with TSM > 0.8
    g) List the three terrestrial planets with the highest TSM
    h) List all the planets that have detected Na in their atmospheres
    i) List the planets that have super-stellar Na abundance
    j) List the planets that have been chracterized by JWST NIRISS and have detected CO2
    k) List all the planets that have any detected species containing Carbon and Oxygen and their and their hosting stars [C/H] and [O/H]
    l) List the period of planets and the metallicities of their hosting stars
    m) List all the O-class stars that host planets
    n) List all the stars older than 10 Gyr from Brewer's dataset and host more than 2 planets
    o) List the metallicities of the stars that are observed by JWST
    p) List the names of the stars that have lithium abundance [Li/H] > 0.3, older than 10 Gyr, and have their planets characterized

4)



6) We plan to load csv files into the database. Data sources:
https://data.sdss.org/datamodel/files/APOGEE_ASPCAP/APRED_VERS/ASPCAP_VERS/allStar.html
https://exoplanetarchive.ipac.caltech.edu/index.html
https://www.stsci.edu/~nnikolov/TrExoLiSTS/JWST/trexolists.html

7) We plan to generate views to look into questions about specific types of planets.
8) Generate spectrum photos using backend code upon user queries.