**Functions**

1. **SQL Intermediate Basics**
2. To see the data types of each column in a table, you can use the PRAGMA statement:

PRAGMA table\_info(facts);

1. Inserting Data into a Table

INSERT INTO facts

VALUES (262, "dq", "DataquestLand", 60000, 40000, 20000, 500000, 100, 50, 10, 20, "2016-02-25 12:00:00", "2016-02-25 12:00:00");

1. Missing Values

SELECT \* FROM facts

WHERE area IS NULL;

insert into facts

values (263, "dq", "DataquestLand", NULL, NULL, 20000, 500000, 100, 50, 10, 20, "2016-02-25 12:00:00", "2016-02-25 12:00:00");

1. Updating rows

UPDATE facts

SET name="New Zealand", code="nz"

WHERE name="Australia"

1. Deleting rows

delete from facts

where name="Canada";

1. Adding and Removing Columns to Table

ALTER TABLE facts

ADD leader text;

ALTER TABLE facts

DROP COLUMN awesomeness;

1. Creating Tables

CREATE TABLE factbook.leaders(

id integer PRIMARY KEY,

name text,

country text

);

1. Creating tables with relations

CREATE TABLE factbook.states(

id integer PRIMARY KEY,

name text,

area integer,

country integer,

FOREIGN KEY(country) REFERENCES facts(id)

);

1. Querying across foreign keys

select \* from landmarks

INNER JOIN facts

ON landmarks.country==facts.id;

1. Left join table

select \* from landmarks

LEFT OUTER JOIN facts

ON landmarks.country==facts.id;

1. Querying from normalized database

SELECT movie FROM nominations

INNER JOIN ceremonies

ON nominations.ceremony\_id == ceremonies.id

WHERE ceremonies.year == 2010 AND nominations.won == 1;

query = 'select ceremonies.year, nominations.movie from nominations INNER JOIN ceremonies ON nominations.ceremony\_id == ceremonies.id where nominations.nominee == "Natalie Portman";'

portman\_movies = conn.execute(query).fetchall()

print(portman\_movies)

1. Join Table

SELECT actors.actor FROM movies

INNER JOIN movies\_actors ON movies.id == movies\_actors.movie\_id

INNER JOIN actors ON movies\_actors.actor\_id == actors.id

WHERE movies.movie == "The Fighter";

kings\_actors = conn.execute('''SELECT actors.actor, movies.movie FROM movies

INNER JOIN movies\_actors

ON movies.id == movies\_actors.movie\_id

INNER JOIN actors ON movies\_actors.actor\_id == actors.id

WHERE movies.movie == "The King's Speech";''').fetchall()

print(kings\_actors)

1. Unique value counts in list

cols = aca\_cols[4:11]

for i in cols:

r = academy[i].value\_counts()

print(i)

print(r)

1. To select first few strings in column of list and convert to integer

academy["Year"] = academy["Year"].str[0:4]

academy["Year"] = academy["Year"].astype("int64")

1. Conditional filtering to select only rows with specified strings

later\_than\_2000 = academy[academy["Year"]> 2000]

award\_categories = ["Actor -- Leading Role",

"Actor -- Supporting Role",

"Actress -- Leading Role",

"Actress -- Supporting Role"]

nominations = later\_than\_2000[later\_than\_2000["Category"].isin(award\_categories)]

additional\_info\_one = final\_nominations["Additional Info"].str.rstrip("'}")

additional\_info\_two = additional\_info\_one.str.split(" {'")

movie\_names = additional\_info\_two.str[0]

characters = additional\_info\_two.str[1]

final\_nominations["Movie"] = movie\_names

final\_nominations["Characters"] = characters

final\_nominations.head()

1. Series method map()

replace = {"NO":0, "YES":1}

nominations["Won?"]= nominations["Won?"].map(replace)

1. Drop() function to drop column

drop = ["Won?","Unnamed: 5", "Unnamed: 6","Unnamed: 7", "Unnamed: 8", "Unnamed: 9", "Unnamed: 10"]

final\_nominations = nominations.drop(drop, axis=1)

1. Exporting to SQLite

import sqlite3

conn = sqlite3.connect("nominations.db")

final\_nominations.to\_sql("nominations", con=conn, index = False)

1. Creating table and inserting data
2. **import** **sqlite3**
3. conn = sqlite3.connect("nominations.db")
4. years\_hosts = [(2010, "Steve Martin"),
5. (2009, "Hugh Jackman"),
6. (2008, "Jon Stewart"),
7. (2007, "Ellen DeGeneres"),
8. (2006, "Jon Stewart"),
9. (2005, "Chris Rock"),
10. (2004, "Billy Crystal"),
11. (2003, "Steve Martin"),
12. (2002, "Whoopi Goldberg"),
13. (2001, "Steve Martin"),
14. (2000, "Billy Crystal"),
15. ]
16. create\_ceremonies = "create table ceremonies (id integer primary key, year integer, host text);"
17. conn.execute(create\_ceremonies)
18. insert\_query = "insert into ceremonies (Year, Host) values (?,?);"
19. conn.executemany(insert\_query, years\_hosts)
20. print(conn.execute("select \* from ceremonies limit 10;").fetchall())
21. print(conn.execute("pragma table\_info(ceremonies);").fetchall())

create\_nominations\_two = '''create table nominations\_two

(id integer primary key,

category text,

nominee text,

movie text,

character text,

won text,

ceremony\_id integer,

foreign key(ceremony\_id) references ceremonies(id));

'''

nom\_query = '''

select ceremonies.id as ceremony\_id, nominations.category as category,

nominations.nominee as nominee, nominations.movie as movie,

nominations.character as character, nominations.won as won

from nominations

inner join ceremonies

on nominations.year == ceremonies.year

;

'''

joined\_nominations = conn.execute(nom\_query).fetchall()

conn.execute(create\_nominations\_two)

insert\_nominations\_two = '''insert into nominations\_two (ceremony\_id, category, nominee, movie, character, won)

values (?,?,?,?,?,?);

'''

conn.executemany(insert\_nominations\_two, joined\_nominations)

print(conn.execute("select \* from nominations\_two limit 5;").fetchall())