Intro to

Oct. 24-26, 2022
Embassy Suites
Concord, NC

Fall 2022 NCASFAA Conference
“Being Spooky Since 1966”
Kris Hatcher EdD, FAAC®

Intro to SQL

- Who am I?
- Why learn about SQL?

Intro to SQL

- What is a spreadsheet?
- What is a database?
- Similarities & differences?
- What is SQL?

Intro to SQL

- What is a spreadsheet?
  - Spreadsheets store data in a table format
  - Spreadsheets have rows and columns

Intro to SQL

- Let’s create a ‘spreadsheet!’

1. Organize the provided data into a spreadsheet by yourself or in a group

2. How did you organize your spreadsheet compared to others in this session?

3. That being said, what happens when…

Intro to SQL

- Situation #1 – The city of “Tallahassee” changes its name to “Bowdenville”

Intro to SQL

- Situation #2 – John Smith adds a second phone number of (904) 248-0085

Intro to SQL

- Situation #3 – Prevent someone from viewing FAFSA information

Intro to SQL

- Situation #4 – You need a list of students whose last name begins with “S”

Intro to SQL

- What is a database?
  - Databases store data
  - Databases can have “rows” and “columns”

Intro to SQL

- Database advantages
    - Data is only stored once
    - Better security is achieved
    - Performance is maximized

Intro to SQL

- Let’s convert our ‘spreadsheet’ into a ‘database’ while addressing the previous situations…

Intro to SQL

- Situation #1 – The city of “Tallahassee” changes its name to “Bowdenville”

Intro to SQL

- Situation #2 – John Smith adds a second phone number of (904) 248-0085

Intro to SQL

- Situation #3 – Prevent someone from viewing FAFSA information

Intro to SQL

- Situation #4 – You need a list of students whose last name begins with “S”

Intro to SQL

- What is SQL?
  - SQL = Structured Query Language
  - Developed by IBM in the 1970s
  - Used to retrieve & manipulate data in a *relational* database

Intro to SQL

- SQL is used to *access* & *modify* information from a relational database:
  - Tables
  - Attributes
  - “Rows”

table

attribute

CITIES

CITY_ID	CITY_NAME	CITY_ACTIVE
1	Tallahassee	Y
2	Panama City	Y
3	Birmingham	Y
4	Atlantis	N

“Rows”

SQL for Beginners

- Basic SQL Instruction (*i.e., “query”*)

```
SELECT  < attribute(s) >

FROM    < table(s) >

WHERE   < condition(s) >
```

table

attribute

CITIES

CITY_ID	CITY_NAME	CITY_ACTIVE
1	Tallahassee	Y
2	Panama City	Y
3	Birmingham	Y
4	Atlantis	N

condition
(*CITY_ACTIVE* = ‘*N*’)

Intro to SQL

CITIES

CITY_ID	CITY_NAME	CITY_ACTIVE
1	Tallahassee	Y
2	Panama City	Y
3	Birmingham	Y
4	Atlantis	N

```
SELECT      CITY_NAME
FROM        CITIES
WHERE       CITY_ACTIVE = ‘Y’
```

Tallahassee
Panama City
Birmingham

Intro to SQL

```
SELECT        PRIMARY_KEY
FROM
        FAFSA_INFORMATION
WHERE         STUDENT_AGI < 1
```

WHERE STUDENT_AGI < 1

000000001
000000002

- *< is a comparison <u>operator</u>*
  - *“Where STUDENT_AGI <u>is less than</u> 1”*

Intro to SQL

```
SELECT      COUNT (LAST_NAME )
FROM        DEMOGRAPHICS
WHERE       LAST_NAME LIKE 'S%'
```

```
3
```

- *A "function" does something, using something*
  - *"Count all the last names"*
  - "Parameters" are listed in parentheses

- *LIKE is a logical operator*
  - *"Where all last names begin with the letter 'S' "*

Intro to SQL

```
SELECT        PRIMARY_KEY
FROM
        FAFSA_INFORMATION
WHERE         CITIZEN IN (‘1’,‘2’)
```

WHERE CITIZEN IN (‘1’,‘2’)

000000001
000000002
000000003
000000004

- *What is the operator?*
  - *IN*
- *What does an “IN” operator do?*
  - *Check for multiple matches, i.e., Where CITIZEN is equal to 1 or 2*

Intro to SQL

```
SELECT      AVG(PARENT_AGI)
FROM        FAFSA_INFORMATION
WHERE       PARENT_AGI IS NOT
NULL
```

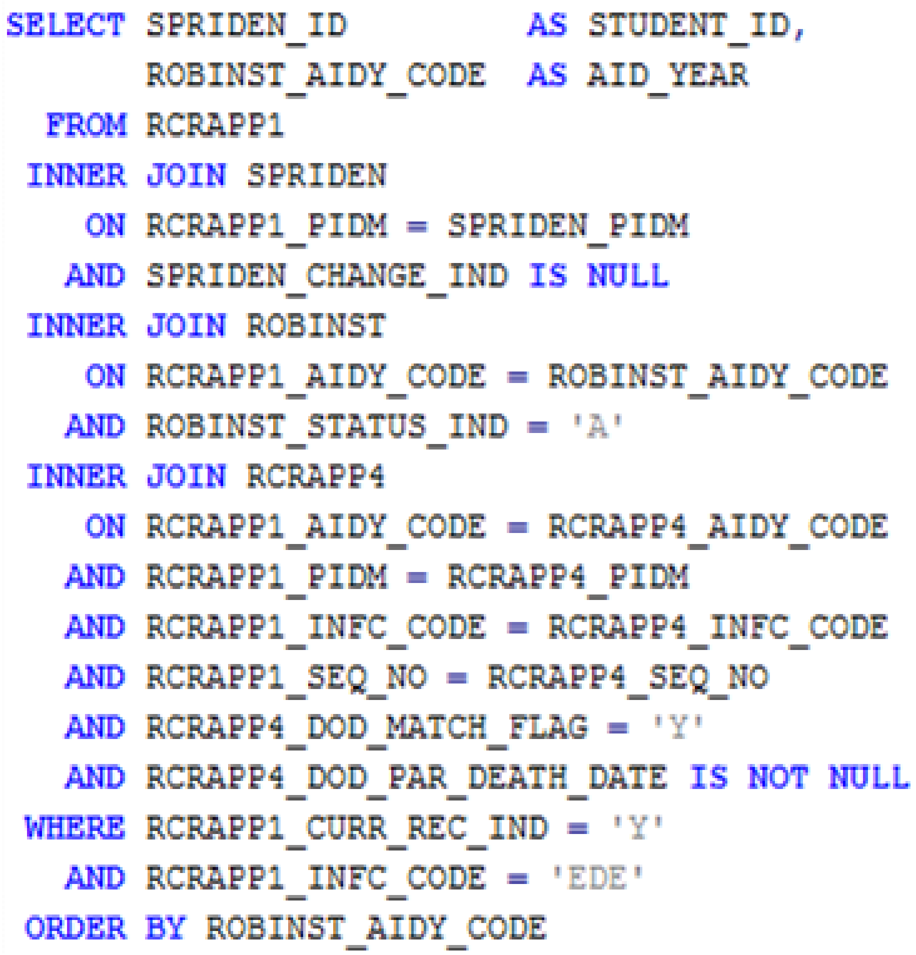
55,000

- *What is the function?*
  - *AVG()*
- *What is the operator?*
  - *IS NOT*
- *What is “NULL?”*
  - *‘Empty’ or ‘No value’ (not zero!)*

Intro to SQL

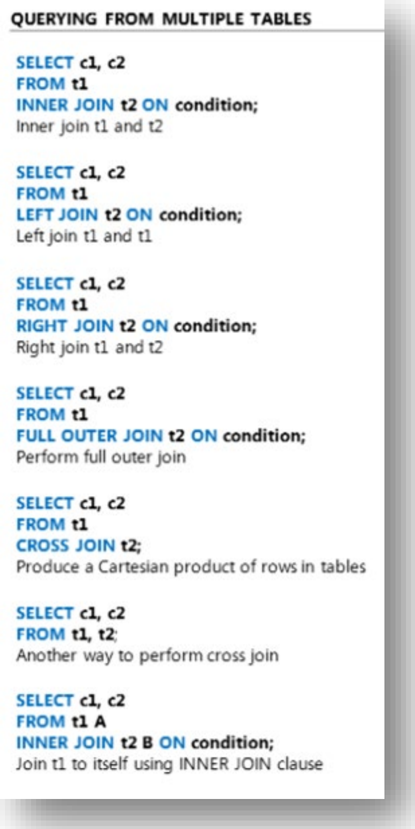
- *Can you guess what this SQL does? (Banner example)*

- *Select students with ISIRs that indicate they may be eligible for Iraq & Afghanistan Service Grant*



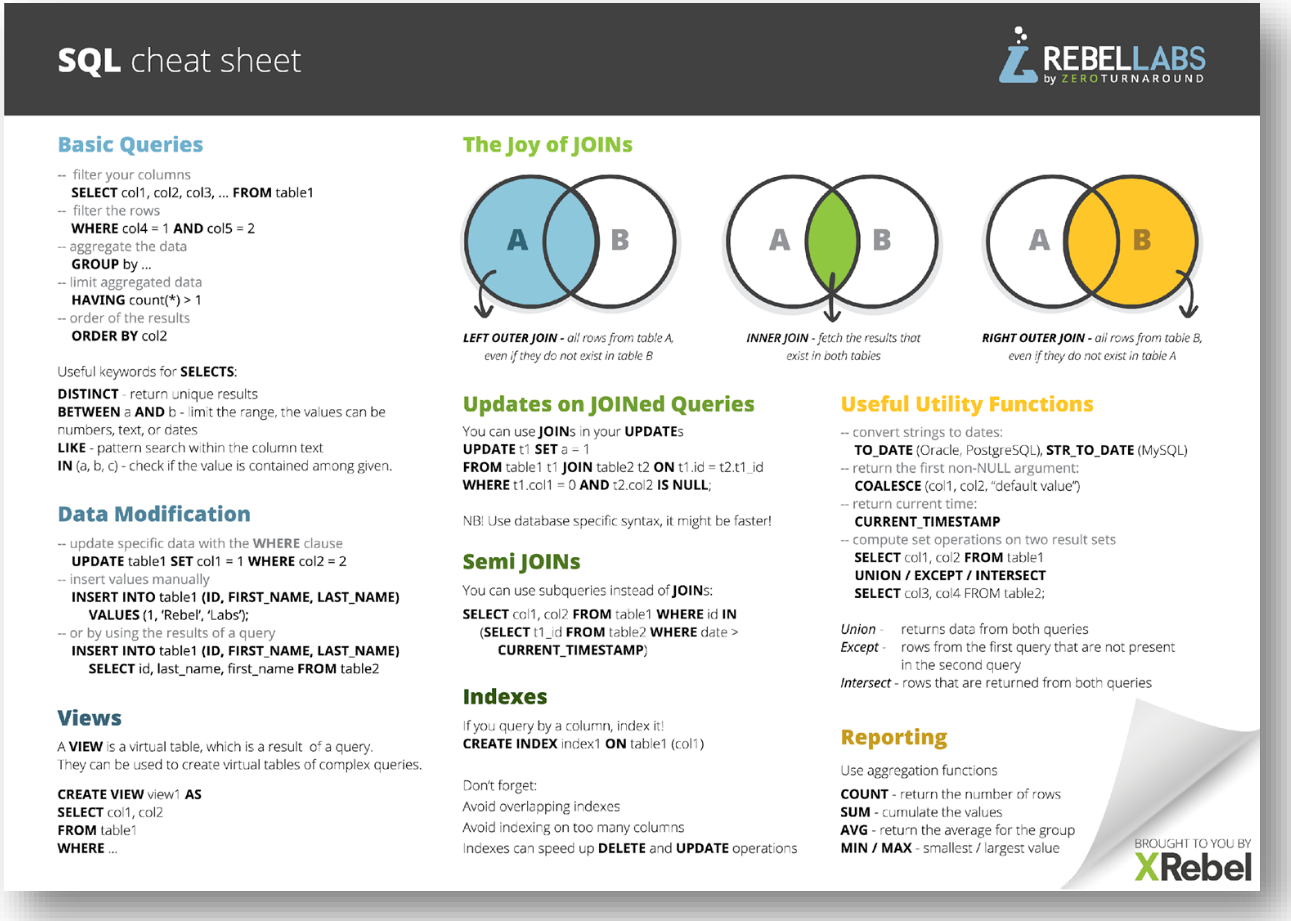
```
SELECT SPRIDEN_ID          AS STUDENT_ID,
       ROBINST_AIDY_CODE   AS AID_YEAR
  FROM RCRAPP1
 INNER JOIN SPRIDEN
    ON RCRAPP1_PIDM = SPRIDEN_PIDM
   AND SPRIDEN_CHANGE_IND IS NULL
 INNER JOIN ROBINST
    ON RCRAPP1_AIDY_CODE = ROBINST_AIDY_CODE
   AND ROBINST_STATUS_IND = 'A'
 INNER JOIN RCRAPP4
    ON RCRAPP1_AIDY_CODE = RCRAPP4_AIDY_CODE
   AND RCRAPP1_PIDM = RCRAPP4_PIDM
   AND RCRAPP1_INFC_CODE = RCRAPP4_INFC_CODE
   AND RCRAPP1_SEQ_NO = RCRAPP4_SEQ_NO
   AND RCRAPP4_DOD_MATCH_FLAG = 'Y'
   AND RCRAPP4_DOD_PAR_DEATH_DATE IS NOT NULL
 WHERE RCRAPP1_CURR_REC_IND = 'Y'
   AND RCRAPP1_INFC_CODE = 'EDE'
 ORDER BY ROBINST_AIDY_CODE
```

Intro to SQL



QUERYING FROM MULTIPLE TABLES

```
SELECT c1, c2
FROM t1
INNER JOIN t2 ON condition;
Inner join t1 and t2

SELECT c1, c2
FROM t1
LEFT JOIN t2 ON condition;
Left join t1 and t1

SELECT c1, c2
FROM t1
RIGHT JOIN t2 ON condition;
Right join t1 and t2

SELECT c1, c2
FROM t1
FULL OUTER JOIN t2 ON condition;
Perform full outer join

SELECT c1, c2
FROM t1
CROSS JOIN t2;
Produce a Cartesian product of rows in tables

SELECT c1, c2
FROM t1, t2;
Another way to perform cross join

SELECT c1, c2
FROM t1 A
INNER JOIN t2 B ON condition;
Join t1 to itself using INNER JOIN clause
```



SQL cheat sheet

REBELLABS by ZEROTURNAROUND

Basic Queries

-- filter your columns
SELECT col1, col2, col3, ... **FROM** table1
-- filter the rows
WHERE col4 = 1 **AND** col5 = 2
-- aggregate the data
GROUP by ...
-- limit aggregated data
HAVING count(*) > 1
-- order of the results
ORDER BY col2

Useful keywords for **SELECTS**:

DISTINCT - return unique results
BETWEEN a **AND** b - limit the range, the values can be numbers, text, or dates
LIKE - pattern search within the column text
IN (a, b, c) - check if the value is contained among given.

Data Modification

-- update specific data with the **WHERE** clause
UPDATE table1 **SET** col1 = 1 **WHERE** col2 = 2
-- insert values manually
INSERT INTO table1 **(ID, FIRST_NAME, LAST_NAME)**
VALUES (1, 'Rebel', 'Labs');
-- or by using the results of a query
INSERT INTO table1 **(ID, FIRST_NAME, LAST_NAME)**
SELECT id, last_name, first_name **FROM** table2

Views

A **VIEW** is a virtual table, which is a result of a query.
They can be used to create virtual tables of complex queries.

CREATE VIEW view1 **AS**
SELECT col1, col2
FROM table1
WHERE ...

The Joy of JOINs

***LEFT OUTER JOIN** - all rows from table A, even if they do not exist in table B*

***INNER JOIN** - fetch the results that exist in both tables*

***RIGHT OUTER JOIN** - all rows from table B, even if they do not exist in table A*

Updates on JOINed Queries

You can use **JOIN**s in your **UPDATE**s
UPDATE t1 **SET** a = 1
FROM table1 t1 **JOIN** table2 t2 **ON** t1.id = t2.t1_id
WHERE t1.col1 = 0 **AND** t2.col2 **IS NULL**;

NB! Use database specific syntax, it might be faster!

Semi JOINs

You can use subqueries instead of **JOIN**s:

SELECT col1, col2 **FROM** table1 **WHERE** id **IN**
(**SELECT** t1_id **FROM** table2 **WHERE** date >
CURRENT_TIMESTAMP)

Indexes

If you query by a column, index it!
CREATE INDEX index1 **ON** table1 (col1)

Don't forget:
Avoid overlapping indexes
Avoid indexing on too many columns
Indexes can speed up **DELETE** and **UPDATE** operations

Useful Utility Functions

-- convert strings to dates:
TO_DATE (Oracle, PostgreSQL), **STR_TO_DATE** (MySQL)
-- return the first non-NULL argument:
COALESCE (col1, col2, "default value")
-- return current time:
CURRENT_TIMESTAMP
-- compute set operations on two result sets
SELECT col1, col2 **FROM** table1
UNION / EXCEPT / INTERSECT
SELECT col3, col4 FROM table2;

Union - returns data from both queries
Except - rows from the first query that are not present in the second query
Intersect - rows that are returned from both queries

Reporting

Use aggregation functions

COUNT - return the number of rows
SUM - cumulate the values
AVG - return the average for the group
MIN / MAX - smallest / largest value

BROUGHT TO YOU BY XRebel

Platinum Level Professional Affiliates

SQL

CFI®

College Foundation, Inc.

Your NC Nonprofit Lender

earnest

College AVE

STUDENT LOANS

SQL

Gold Level Supporters

ADVANTAGE
Education Loan

Inceptia
A DIVISION OF NSLP

sallie mae®

PNC BANK

Citizens One
Student Loans®

Silver Level Professional Affiliates

CollegeRaptor

MPOWER Financing

Ascent

Loans that power bright futures.

DISCOVER®

STUDENT LOANS

Ascent

Loans that power bright futures.

SoFi

CUSTOM CHOICE®

union federal®

Powered by

cognition

FINANCIAL

nextgen

WEB SOLUTIONS

Bronze Level Professional Affiliates

ECMC

Questions?

https://www.w3schools.com/sql/

https://www.youtube.com/c/cs50/

kristohper_hatcher@hotmail.com

@HatchEdD on Twitter