SQL Basics

* SELECT
  + Capitalizing not required; easier read by distinguishing column names from clauses
  + Semicolon at end of query
  + DISTINCT – unique; avoid duplicate rows
  + NULLs – reduce possibilities of null. Best to use blanks or 0, but if it skews results use null
    - IS NULL/IS NOT NULL – test a column for nulls
  + AS – alias; write Expressions as alias to read easier
  + CASE - statement goes through conditions and returns a value when the first condition is met (like an if-then-else statement). once a condition is true, it will return the result. If no conditions are true, it returns the value in the ELSE clause. If there is no ELSE part and no conditions are true, it returns NULL.

SELECT patient\_id, height,

CASE

WHEN height>175 THEN ‘height greater than 175’

WHEN height>175 THEN ‘height greater than 175’

ELSE ‘height is under 175’

|  |  |
| --- | --- |
| 'a%' | start with "a" |
| ‘%a' | end with "a" |
| '%or%' | have "or" in any position |
| '\_r%' | have "r" in the second position |
| 'a\_%' | start with "a" and at least 2 characters |
| 'a\_\_%' | start with "a" and at least 3 characters |
| 'a%o' | start with "a" and ends with "o" |

END AS ‘…’

* WHERE – case sensitive; use quotes for strings
  + AND/OR
  + =, !=, <,>,<=
  + IN/NOT IN() - in or not in a list
  + BETWEEN… AND – within range
  + NOT BETWEEN – not within range
  + LIKE/NOT LIKE – return similar, not case sensitive
    - % and \_ use w LIKE/NOT LIKE only
* GROUP BY – grouping rows with same value in column specified
  + Must use group by with aggregate function or it will run on whole set & return 1 value
    - Give aggregate function an alias; easier to read
    - Aggregate functions (COUNT, MIN, MAX, SUM, AVG, ROUND)
* HAVING – filtering group by rows; *same constraints as WHERE clause*
* ORDER BY – sort results
  + ASC/DESC
  + LIMIT clause– reduce # rows returned
  + OFFSET clause – where to begin counting rows
* JOIN – combine separate tables using unique key
  + INNER JOIN (JOIN)– matches rows from both tables which have same key (**ON**)

SELECT column, another\_table\_column, …

FROM mytable

**INNER JOIN** another\_table

ON mytable.id = another\_table.id

WHERE *conditions(s)*

ORDER BY *column, ..*ASC/DESC

LIMIT num\_lmit OFFSET num\_offset;

* + LEFT JOIN/ RIGHT JOIN/ FULL JOIN - may need to write additional logic to deal w NULLs

SELECT column, another\_table\_column, …

FROM mytable

**INNER/LEFT/RIGHT/FULL JOIN** another\_table

ON mytable.id = another\_table.matching\_id

WHERE *conditions(s)*

ORDER BY *column, ..*ASC/DESC

LIMIT num\_lmit OFFSET num\_offset;

* Examples of full Query

SELECT DISTINCT column, AGG\_FUNC(*column\_or\_expression*), …

FROM mytable

JOIN another\_table

ON mytable.column = another\_table.column

WHERE *constraint\_expression*

GROUP BY column

HAVING *constraint\_expression*

ORDER BY *column* ASC/DESC

LIMIT *count* OFFSET *COUNT*;

INSERT INTO – statement to insert new data into table

INSERT INTO mytable

VALUES (value\_or\_expr, another\_value\_or\_expr, …),

(value\_or\_expr\_2, another\_value\_or\_expr\_2, …)

UPDATE – statement to add existing data, must match data type in schema; *use w/ WHERE*

UPDATE mytable

**SET** column = value\_or\_expr,

other\_column = another\_value\_or\_expr,

WHERE condition;

DELETE FROM – statement to delete data from a table; *use WHERE clause*

DELETE FROM mytable

WHERE condition;

CREATE TABLE

CREATE TABLE **IF NOT EXISTS** mytable (

column *DataType* *TableConstraint* DEFAULT *default\_value*,

another\_column *DataType* *TableConstraint* DEFAULT *default\_value*,

);

Data Types

* Integer, Boolean
* Float, double, real – more precise numeric values; fraction, decimal, etc
* Character/varchar, text
* Date, datetime
* Blob –

Constraints

|  |  |
| --- | --- |
| PRIMARY KEY | - unique value |
| AUTOINCREMENT | - for integer values; value automatically filled |
| UNIQUE | - value must be unique |
| NOT NULL | - inserted value cannot be NULL |
| CHECK (expression) | - test weather values inserted are valid |
| FOREIGN KEY | - |

Schema example:

CREATE TABLE movies (

id INTEGER PRIMARY KEY,

title TEXT,

director TEXT,

year INTEGER,

length\_minutes INTEGER

);

ALTER TABLE – statement to add, remove, modify columns & table constraints

* Adding columns

ALTER TABLE mytable

**ADD** column *DataType* *OptionalTableConstraint*

DEFAULT default\_value;

* Removing columns

ALTER TABLE mytable

**DROP** column\_to\_be\_deleted;

* Renaming table

ALTER TABLE mytable

**RENAME TO** new\_table\_name;

DROP TABLE – statement removes all data & metadata ; also removes schema from database entirely

* if you have another table that is dependent on deleted table, then you will have to either update all dependent tables first to remove the dependent rows or tables entirely.

DROP TABLE IF EXISTS mytable;

SQL Project: Murder Mystery

Overview: Murder occurred in SQL City on 1/15/ 2018

1. Find all crime reports that occurred in SQL city on 1/15/18

SELECT \*

FROM crime\_scene\_report

WHERE city = 'SQL City' AND date = '20180115';

* Provides date, type, city, description of all crimes on that day.
* There are 2 witnesses, 1st witness lives at the last house on "Northwestern Dr"
* The second witness, named Annabel, lives somewhere on "Franklin Ave".

1. Track down witness 1 and Find all information on them using information from the report

SELECT \*

FROM person

WHERE address\_street\_name = 'Northwestern Dr'

ORDER BY address\_number DESC

LIMIT 1;

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 14887 | Morty Schapiro | 118009 | 4919 | Northwestern Dr | 111564949 |

1. Find all information on witness 2

SELECT \*

FROM person

WHERE address\_street\_name = 'Franklin Ave' AND name LIKE '%Annabel%';

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 16371 | Annabel Miller | 490173 | 103 | Franklin Ave | 318771143 |

1. Check interview notes of both witnesses for more details

SELECT \*

FROM interview

WHERE person\_id = '14887' OR person\_id = '16371';

| **person\_id** | **transcript** |
| --- | --- |
| 14887 | I heard a gunshot and then saw a man run out. He had a "Get Fit Now Gym" bag. The membership number on the bag started with "48Z". Only gold members have those bags. The man got into a car with a plate that included "H42W". |

|  |  |
| --- | --- |
| 16371 | I saw the murder happen, and I recognized the killer from my gym when I was working out last week on January the 9th. |

1. Look up the suspect using all the information the witnesses gave on him

SELECT \*

FROM get\_fit\_now\_member

JOIN get\_fit\_now\_check\_in ON get\_fit\_now\_member.id = get\_fit\_now\_check\_in.membership\_id

WHERE membership\_id LIKE '48Z%'

AND membership\_status = 'gold'

AND check\_in\_date = '20180109';

| **id** | **person\_id** | **name** | **membership\_start\_date** | **membership\_status** | **membership\_id** | **check\_in\_date** | **check\_in\_time** | **check\_out\_time** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 48Z7A | 28819 | Joe Germuska | 20160305 | gold | 48Z7A | 20180109 | 1600 | 1730 |
| 48Z55 | 67318 | Jeremy Bowers | 20160101 | gold | 48Z55 | 20180109 | 1530 | 1700 |

1. Witness mentioned suspects partial licensed plate. Match the license plate to either of the 2 suspects.

SELECT person.name, drivers\_license.\*

FROM drivers\_license

JOIN person ON drivers\_license.id = person.license\_id

WHERE plate\_number like '%H42W%' AND gender = 'male';

| **name** | **id** | **age** | **height** | **eye\_color** | **hair\_color** | **gender** | **plate\_number** | **car\_make** | **car\_model** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tushar Chandra | 664760 | 21 | 71 | black | black | male | 4H42WR | Nissan | Altima |
| Jeremy Bowers | 423327 | 30 | 70 | brown | brown | male | 0H42W2 | Chevrolet | Spark LS |

1. Jeremy is the killer but find out the motive. Find Jeremy’s interview.

SELECT transcript, name

FROM interview

JOIN person ON interview.person\_id = person.id

WHERE name = 'Jeremy Bowers';

| **transcript** | **name** |
| --- | --- |
| I was hired by a woman with a lot of money. I don't know her name but I know she's around 5'5" (65") or 5'7" (67"). She has red hair and she drives a Tesla Model S. I know that she attended the SQL Symphony Concert 3 times in December 2017. | Jeremy Bowers |

1. Search the description of the woman

SELECT person.id, name, height, hair\_color, ssn, car\_model

FROM drivers\_license

JOIN person ON drivers\_license.id = person.license\_id

WHERE height BETWEEN 65 AND 67

AND hair\_color = 'red'

AND car\_make = 'Tesla'

AND car\_model = 'Model S'

| **id** | **name** | **height** | **hair\_color** | **ssn** | **car\_model** |
| --- | --- | --- | --- | --- | --- |
| 78881 | Red Korb | 65 | red | 961388910 | Model S |
| 90700 | Regina George | 66 | red | 337169072 | Model S |
| 99716 | Miranda Priestly | 66 | red | 987756388 | Model S |

1. Find the mastermind of the crime by looking up event and date

SELECT name

FROM facebook\_event\_checkin

JOIN person ON facebook\_event\_checkin.person\_id = person.id

WHERE event\_name LIKE 'SQL Symphony Concert'

AND date LIKE '201712\_\_'

| **name** |
| --- |
| Miranda Priestly |

AND ssn = '961388910'

OR ssn = '337169072'

OR ssn = '987756388'

GROUP BY name;