UNIVERSITY OF
SAN FRANCISCO

Master of Science
in Analytics

SQL

Interview Skills

SQL Questions

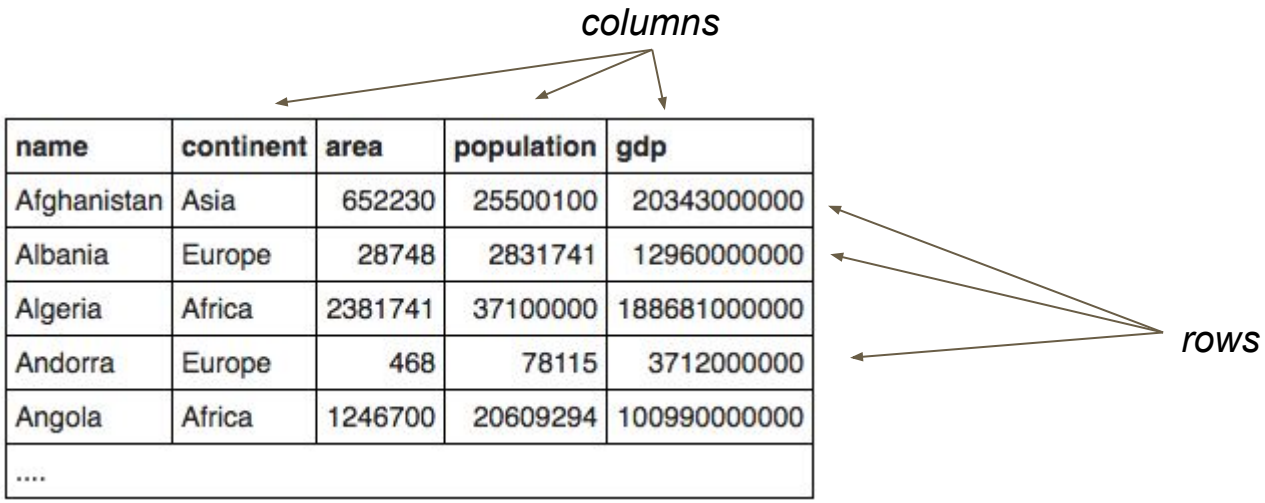
- Likely to get example as a table or set of tables
  - Treat example as an example (not WYSIWYG)
  - Do not assume example is complete (eg. may be missing countries, continents, etc.)

name	continent	area	population	gdp
Afghanistan	Asia	652230	25500100	20343000000
Albania	Europe	28748	2831741	12960000000
Algeria	Africa	2381741	37100000	188681000000
Andorra	Europe	468	78115	3712000000
Angola	Africa	1246700	20609294	100990000000
....				

- Likely to be asked ONLY about SELECT statements
- Highly unlikely to be asked
  - CREATE table / DROP table
  - Users, DB security, etc.

Tables

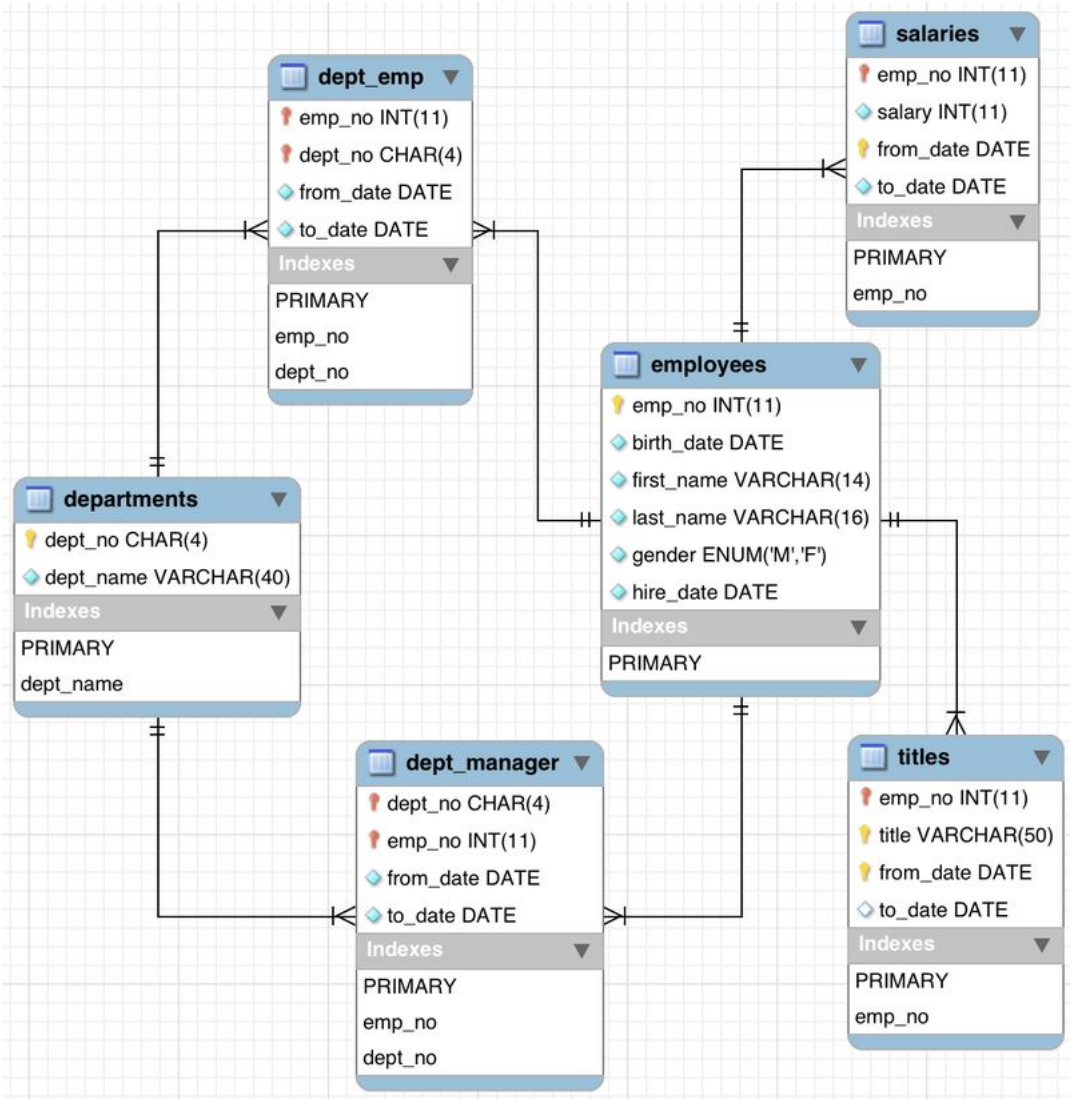
- Database contains one or more tables
  - Table is identified by name
  - Table contains records (rows) of data
  - Each row is broken up into columns — all rows in a table have the same columns



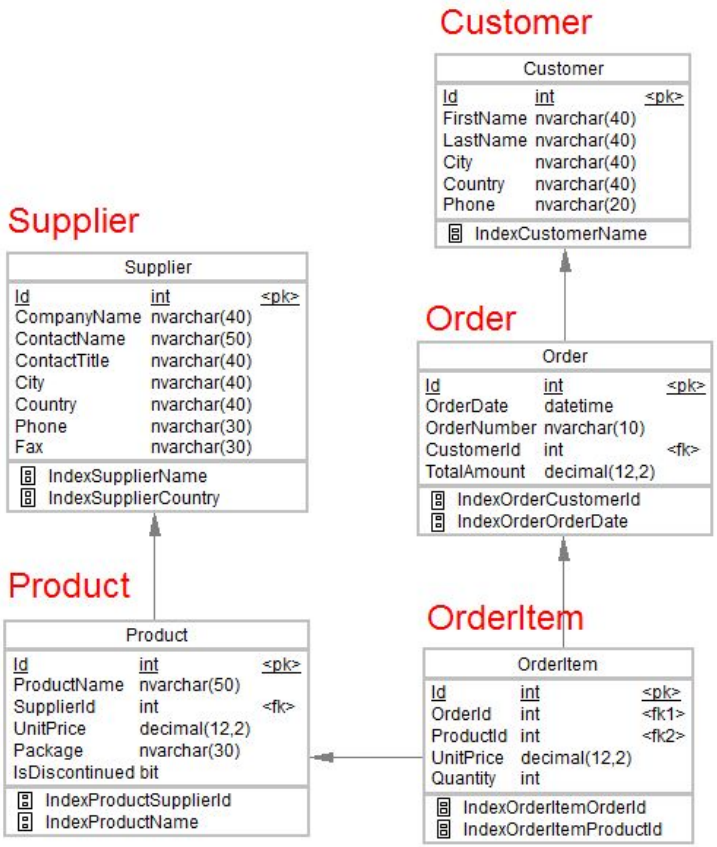
name	continent	area	population	gdp
Afghanistan	Asia	652230	25500100	20343000000
Albania	Europe	28748	2831741	12960000000
Algeria	Africa	2381741	37100000	188681000000
Andorra	Europe	468	78115	3712000000
Angola	Africa	1246700	20609294	100990000000
....				

- Syntax
  - SQL keywords are not case sensitive (i.e. select == SELECT)
  - Table names and column names are case sensitive
  - Semicolons are required as ending to SQL statements on SOME systems, not all

You May See a Schema



Schema from https://dev.mysql.com/doc/employee/en/sakila-structure.html



Schema from http://www.dofactory.com/sql/sample-database

Data Types

- Each column must have a name and type
  - Columns may have additional constraints defined when table it created/altered
  - Each SQL implementation may contain slightly different names
- Most common types:

Data Type	Stores	Notes
BOOLEAN	TRUE/FALSE	Also: BIT/BYTE; Values may be: Yes/No
INTEGER	A whole number; default max precision ~ 10 digits	Also: BIGINT, INT, NUMBER, SMALLINT
FLOAT	Floating-point number; default mantissa ~ 16	Also: NUMBER, NUMERIC, REAL
DECIMAL	Floating-point number; may specify precision — eg. FLOAT(5, 2)	Also: CURRENCY, MONEY
CHAR	Fixed-length string	Also: CHARACTER
VARCHAR	Variable-length string	

Relationships

- Primary key
  - A table usually has a (primary) key
  - Key uniquely identifies a row in the table
  - Often key is artificial (often called “ID”) — i.e. unrelated to the data in the row
- Foreign key
  - Primary key in one table copied to another table
  - Defines relationship between tables

actor

name	date_of_birth	place_of_birth	...
John Boyega	1992-03-17	London, UK	...
Daisy Ridley	1992-04-10	London, UK	...
Adam Driver	1983-11-19	San Diego, CA	...

acts_in

title	release_date	name	...
Star Wars: The Force Awakens	2015-12-14	John Boyega	...
Attack the Block	2011-07-29	John Boyega	...
Star Wars: The Force Awakens	2015-12-14	Daisy Ridley	...
Star Wars: The Force Awakens	2015-12-14	Adam Driver	...

movie

title	release_date	running_time	...
Star Wars: The Force Awakens	2015-12-14	135 minutes	...
Attack the Block	2011-07-29	88 minutes	...

INSERT Statement

- Places data into table
- Example table = muni_train:
- Syntax

id	line	direction
1	J	inbound
2	K	inbound

```
insert into TABLE values (VAL1, VAL2, ...);
```

  - Assumes values in order of columns listed; better to state columns explicitly

```
insert into TABLE (COL1, COL2, ...) values (VAL1,
VAL2, ...);
```

- Example (adds outbound J)

```
insert into muni_train (id, line, direction) values
(3, 'J', 'outbound');
```

DELETE Statement

- Removes data from table
- Example table = muni_train:
- Syntax

id	line	direction
1	J	inbound
2	K	inbound

```
delete from TABLE where COL=VAL;
```

  - Assumes COL uniquely identifies a row in the table
- Example (removes inbound K)

```
delete from muni_train where id = 2;
```

UPDATE Statement

- Changes data in table
- Example table = muni_train:
- Syntax

id	line	direction
1	J	inbound
2	K	inbound

```
update TABLE set COL1=VAL1, COL2=VAL2 where COL=VAL;
```

  - Assumes COL uniquely identifies a row in the table
- Example (removes inbound K)

```
update muni_train set line='J', direction='outbound'
where id = 2;
```

SELECT Statement

- Retrieves data in table
- Example table = muni_train:
- Simple syntax

id	line	direction
1	J	inbound
2	K	inbound

```
select COL1, COL2 from TABLE;
```

- Example (retrieves all ids, lines & directions)

```
select * from muni_train;
```

- Example (retrieves all lines & directions)

```
select line, direction from muni_train;
```

SELECT WHERE

id	line	direction
1	J	inbound
2	K	inbound

- WHERE clause
  - Used to extract only records which fulfil criterion
  - In other words, to filter records
- Simple syntax

```
select COL1, COL2 from TABLE where OPERATOR VALUE;
```

- Example (retrieves inbound lines)

```
select * from muni_train where direction='inbound';
```

- Example (retrieves records with IDs less than 10)

```
select line, direction from muni_train where id < 10;
```

- May combine multiple clauses with AND, OR

```
select * from muni_train where id < 10 and line='K';
```

Like Operator

name	continent	area	population	gdp
Afghanistan	Asia	652230	25500100	20343000000
Albania	Europe	28748	2831741	12960000000
Algeria	Africa	2381741	37100000	188681000000
Andorra	Europe	468	78115	3712000000
Angola	Africa	1246700	20609294	100990000000
....				

- Like Operator
  - Searches for a specified pattern in a column
  - Used in a WHERE clause
- Simple syntax

```
select COL1 from TABLE where COL1 like PATTERN;
```

- Wildcards (string / varchar types):
  - % = Any number of characters
  - _ = Any single character
  - [CHARLIST] = Set (or range) of characters to match
  - [^CHARLIST] = Set of characters NOT to match (also: [!CHARLIST]
- Example (retrieves countries ending in “land”)

```
select name from country where name like '%land';
```

In Operator

- In Operator
  - Specifies particular values
  - Used in a WHERE clause
- Simple syntax

name	continent	area	population	gdp
Afghanistan	Asia	652230	25500100	20343000000
Albania	Europe	28748	2831741	12960000000
Algeria	Africa	2381741	37100000	188681000000
Andorra	Europe	468	78115	3712000000
Angola	Africa	1246700	20609294	100990000000
....				

```
select COL1 from TABLE where COL1 in (VAL1, VAL2);
```

- Example (retrieves all American, Asian countries)

```
select name from country where continent in ('Asia',
'North America', 'South America');
```

SELECT DISTINCT

- Possible duplicates
  - Column (even table) may contain duplicate values
  - Sometimes you need unique (different/distinct) values
- Simple syntax

```
select distinct COL1, COL2 from TABLE;
```

- Example (retrieves unique lines)

```
select distinct line from muni_train;
```

id	line	direction
1	J	inbound
2	K	inbound

Order By

- Sorting
  - By default, results from table are not sorted / ordered
  - Use “order by” keyword to impose ordering on results
  - Ascending order is default
- Simple syntax

id	line	direction
1	J	inbound
2	K	inbound

```
select distinct COL1, COL2 from TABLE order by COL1 DESC, COL2;
```

- Example (sorts results by lines, directions)

```
select line, direction from muni_train order by line, direction;
```

Alias

- Alias
  - Temporarily rename a column (or table)
  - Uses “as” keyword
- Simple syntax

```
select COL as ALIAS from TABLE;
```

- Example (selects lines, names the column “train”)

```
select line as train from muni_train;
```

- Example (name a concatenation)

```
select direction+' '+ line as train from muni_train;
```

```
select CONCAT(direction, ' ', line) as train from
muni_train;
```

id	line	direction
1	J	inbound
2	K	inbound

Limit / Rownum / Top

- Specify number of records to return
  - Useful (for testing?) vs. large tables
  - Note: not all databases support TOP clause
  - On MySQL/Oracle: LIMIT/ROWNUM
- Simple syntax

```
select top NUMBER|PERCENT * from TABLE;

select * from TABLE limit NUMBER;

select * from TABLE rownum <= NUMBER;
```

- Example (only 5 rows)

```
select top 5 * from muni_train;

select * from muni_train limit 5;

select * from muni_train rownum <=5;
```

id	line	direction
1	J	inbound
2	K	inbound

Joins, Inner Join

- Combine rows from two or more tables
  - Based on a common field between tables
  - Different types: INNER, LEFT, RIGHT, FULL
- Example
  - Two tables: order, customer
  - Field order.customer_id ⇒ customer.id

order

id	date	customer_id
1	2016-01-01	27
2	2016-01-01	18
3	2016-01-02	573
4	2016-01-03	18

customer

id	first_name	last_name
18	Sylvia	Poggioli
27	Lakshmi	Singh
573	Ryan	Cole

```
select customer.id, customer.last_name, order.id,
order.date from order inner join customer on
order.customer_id = customer.id;
```

Left Join

- What it does
  - Returns all rows from left table and all matching rows in right table
  - Returns NULL values when there's no match in the right table
- Example
  - Two tables: order, customer
  - Field order.customer_id ⇒ customer.id

order

id	date	customer_id
1	2016-01-01	27
2	2016-01-01	18
3	2016-01-02	573
4	2016-01-03	18

customer

id	first_name	last_name
18	Sylvia	Poggioli
27	Lakshmi	Singh
573	Ryan	Cole

```
select customer.last_name, order.id, order.date from
order left join customer on order.customer_id =
customer.id;
```

Right Join

- What it does
  - Returns all rows from right table and all matching rows in left table
  - Returns NULL values when there's no match in the left table
- Example
  - Two tables: order, customer
  - Field order.customer_id ⇒ customer.id

order

id	date	customer_id
1	2016-01-01	27
2	2016-01-01	18
3	2016-01-02	573
4	2016-01-03	18

customer

id	first_name	last_name
18	Sylvia	Poggioli
27	Lakshmi	Singh
573	Ryan	Cole

```
select customer.last_name, order.id, order.date from
order right join customer on order.customer_id =
customer.id;
```

Full Join

- What it does
  - Returns all rows from all tables
  - Returns NULL values when there's no match in one table or the other
- Example
  - Two tables: order, customer
  - Field order.customer_id ⇒ customer.id

order

id	date	customer_id
1	2016-01-01	27
2	2016-01-01	18
3	2016-01-02	573
4	2016-01-03	18

customer

id	first_name	last_name
18	Sylvia	Poggioli
27	Lakshmi	Singh
573	Ryan	Cole

```
select * from order full outer join customer on
order.customer_id = customer.id;
```

Union

- What it does
  - Combines the result set of two or more select statements
  - Each statement must have same number of columns, similar (identical?) data types
  - Columns must be in the same order
- Example
  - Two unrelated tables: customer, supplier
  - Objective: retrieve all names in the tables

supplier

id	bus_name	city
25	Keen	Portland
62	ILM	San Francisco
50	Pfizer	Skokie
4	Publix	Miami

customer

id	first_name	last_name
18	Sylvia	Poggioli
27	Lakshmi	Singh
573	Ryan	Cole

```
select last_name from customer union select bus_name
from supplier;
```

Scalar Functions

Function	Description
CONCAT(x, y)	Concatenates fields x and y
UCASE(x)	Converts x to upper case (sometimes called “UPPER”)
LCASE(x)	Converts x to lower case (sometimes called “LOWER”)
LEN(x)	Returns the length of string, x
ROUND(x, y)	Rounds x to y places after decimal; round(214.23, -2) ⇒ 200.00

- Example

```
select UCASE(last_name) from customer;
```

Aggregate Functions

Function	Description
AVG(x)	Returns the average value of column x
COUNT(x)	Returns the number of rows in column x
MAX(x)	Returns the largest value in column x
MIN(x)	Returns the smallest value in column x
SUM(x)	Returns the sum of column x

- Example

```
select sum(quantity) as total_item_count from order;
```

Group By

- What it does
  - Used in aggregate function
  - Groups result set by one or more columns
- Example
  - Two tables: customer, order
  - Objective: retrieve the number of orders for each customer

order

id	date	customer_id
1	2016-01-01	27
2	2016-01-01	18
3	2016-01-02	573
4	2016-01-03	18

customer

id	first_name	last_name
18	Sylvia	Poggioli
27	Lakshmi	Singh
573	Ryan	Cole

```
select first_name, last_name, count(order.id) from
customer left join order on order.customer_id =
customer.id group by customer.id;
```

Having

- What it does
  - Used to limit result set according to aggregate function
  - Often used with GROUP BY
- Example
  - Two tables: customer, order
  - Objective: retrieve the number of orders for each customer with at least 10 orders

order

id	date	customer_id
1	2016-01-01	27
2	2016-01-01	18
3	2016-01-02	573
4	2016-01-03	18

customer

id	first_name	last_name
18	Sylvia	Poggioli
27	Lakshmi	Singh
573	Ryan	Cole

```
select last_name, count(order.id) from customer left
join order on order.customer_id = customer.id group by
customer.id having count(order.id) > 10;
```

Resources

- SQLZoo has explanations and practices, for example:
  - Guest House
  - AdventureWorks
- The sandbox at dofactory