SQL - FULL

Back End - responsible for the business logic, codes that will send the right request to the DatBase
Database - there is no logic, just information

We took the key word from the Front End → back end taking those keywords and deciding what is this, what belongs to, where is coming from, what is the business logic for it → and moving to the database (give me all the products that related to the book and return it to the UI.

BackEnd testing tools:

- API
- Oracle DataBase
- Restful API
- Java JDBC
- SQL
- Postman

What is Data:

- Piece of information
- For example:
  - Account number → 123
  - Account Type → Checking
  - User Firstname → John

Database:

- Systematic collection of data
- Support storage and manipulation of the data
- Make data management easy

IQ: **What is the primary key?**. One column that makes each column unique, it can not be null and can not be duplicated. THIS IS A Unique identifier.

Primary key	DEPARTMENTS
department_id	departments_name
10	HR
20	IT
30	Finance

RELATION OF DATABASE SYSTEM

RELATION (CONNECTION) BETWEEN TABLES

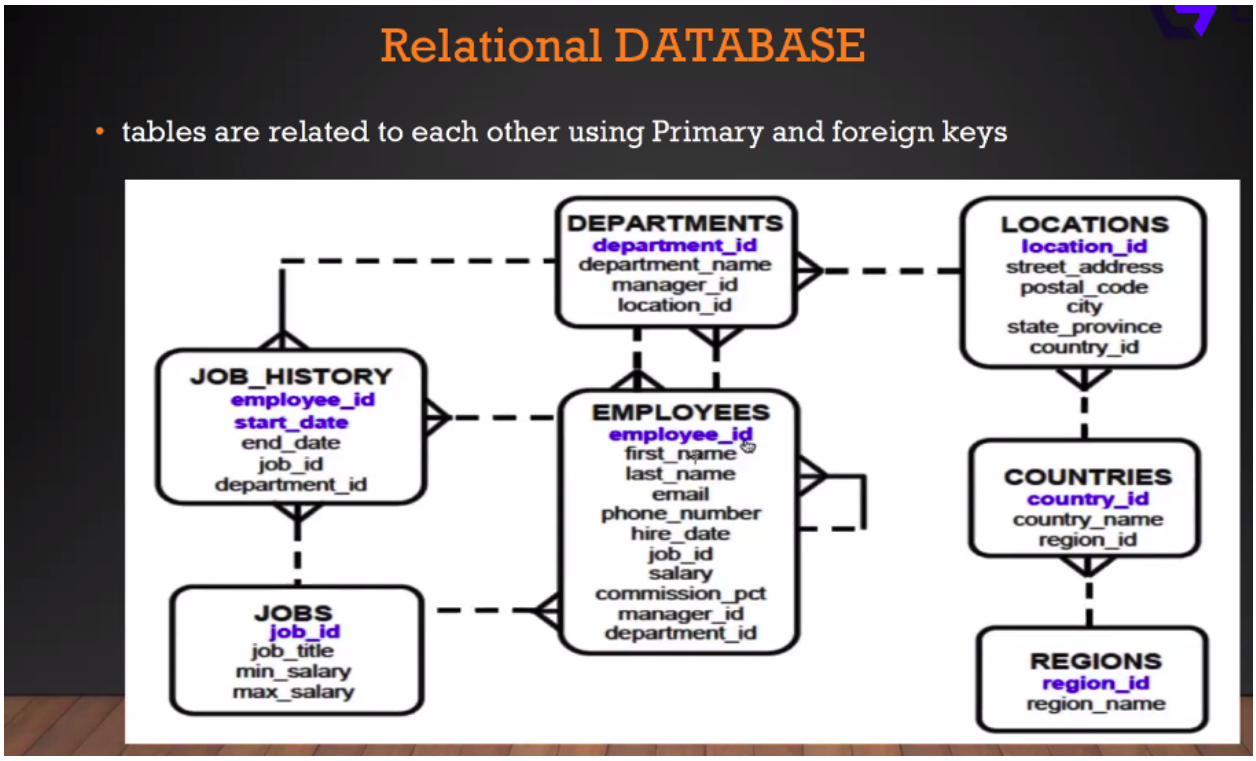
Primary key	EMPLOYEES				foreign key
emp_id	first_name	last_name	title	salary	department_id
1	Mike	Smith	tester	130000	20
2	Murodil	Sdet	developer	200000	20
3	John	Doe	accountant	70000	30
	123213	123asd	manage	ten thousand	
4	Mike	Smith	tester	130000	20

Primary key	DEPARTMENTS
department_id	departments_name
10	HR
20	T
30	Finance

When you have one table with unique primary key and in another table you have a reference there, in the table where is the reference can be duplicates

If you use the ***primary key*** in one table in another table → another table is using ***foreign key*** (can be duplicate and null) because there might be employees that are not assigned to any department yet.

Table can have one primary key and multiple foraign key



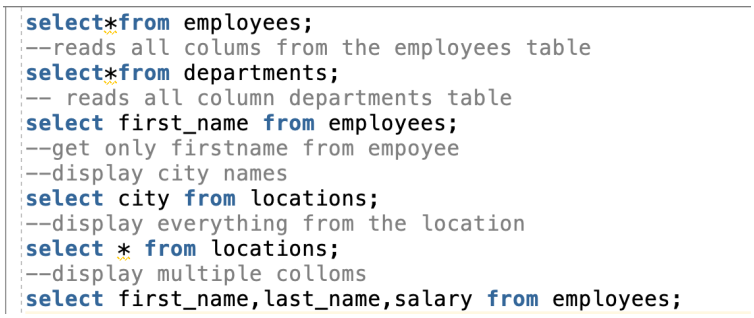
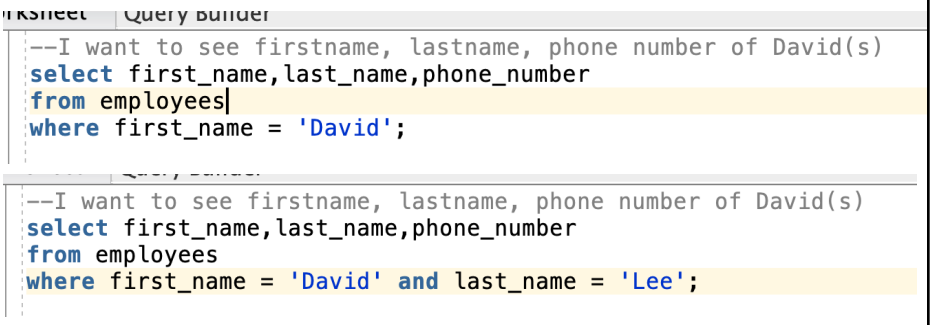
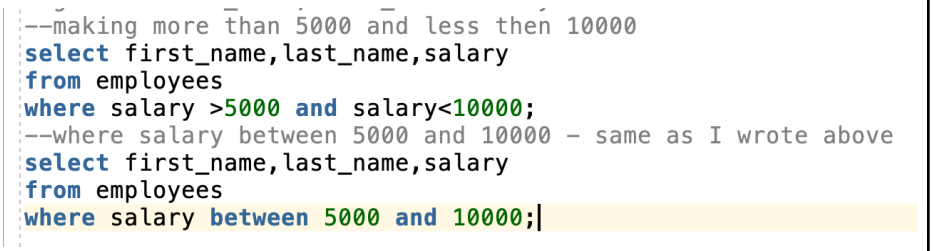
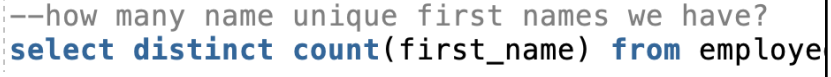
BLUE ON THE PICTURE ABOVE IS PRIMARY KEY AND FOREIGN KEY IS BLACK

- (RDBMS) - RELATIONAL DATABASE MANAGEMENT SYSTEM
- All RDBMS using SQL language
- Relational Database - tables are related to each other using Primary and foreign keys

EC2 machine - we always instal the database inside

SQL - Structured query language
SQL is a language that is used to work with Databases and manipulate data
SQL - is not testing tool

SQL language:	
- Select	- not manipulating any data, it will just show the result in the query

	
Choche the specific information from the DataBase	
- “*”	- all (select * - I wanna select everything from the table)
- Employees	- name of the table(can vary)If you want to run specific line, first click on the line then Run
--	- means comment
- Between	Get information in between specific ranges . And is used just for adding 
- Distinct	- Remove duplicates from the view query (How to understand if it takes the first or last duplicate? - it take the last one) 
- in	Get for me specific one

	``--get me all info where employee_id 135,176,154,129 select * from employees where employee_id=129 or employee_id=135 or employee_id=154 or employee_id=176; --same result in keyword select * from employees where employee_id in(135,176,154,129);```
count(*) or count(name)	Retrieve one cell and give the result, number of the result. - 35 lines is a result ```--how many employee working as IT_PROG or SA_REP select count(*) from employees where job_id in('IT_PROG','SA_REP');``` Unique names ```--how many name unique first names we have? select count(distinct first_name) from employees; select distinct count(first_name) --wrong```
Order by … asc/desc Default order is asc.	
Desc : - 9-0 - Z-A	```--get me all employees information based on who is making --more salary to low select * from employees order by salary desc;```
Asc: - 0-9 - A-Z	```--low to more select * from employees order by salary asc; --get me all emp inf order by alphabetical based on firstname select * from employees order by first_name asc;```
Like: - If letter is in the middle - case sensitive	
Like + % (% any sequence of character)	```--get me all empoyees whose first name starts with C select * from employees where first_name like 'C%';``` so that % means that there could be anywhere from 1 to infinity characters after the first letter

Like + _
_ - represent the character

```
--get me all empoyees whose first name starts with C
select * from employees
where first_name like 'C____';
```

```
--get me all empoyees whose first name starts with C
select * from employees
where first_name like 'C_r%'; -- c_(one empty)r%(and anything after it)
-- where first_name like 'C%'; - any sequance of the character
-- where first_name like 'C____'; - each _ is one letter

-- get me 5 letter first names where the middle char is z
select * from employees
where first_name like '__u__'; --in the middle letter has to be lowecase

--get me first name where second char is u
select * from employees
where first_name like '_u%';
```

AGGREGATE Function

min

max

From a lot of lines it will return one

```
--find minimum/max salary
select min(salary) from employees; -- 1200
select max(salary) from employees; --24000
```

avg (average

It will take the whole amount of all the salaries and divide by the number of employees to get average

```
--find avarage salary
select avg(salary) from employees;  --6461.12432342124
--round
select round(avg(salary)) from employees; --6462
--round with desimal
select round(avg(salary),2) from employees; --6462
```

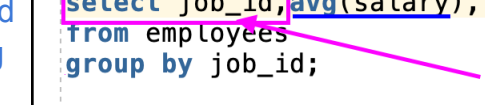
sum

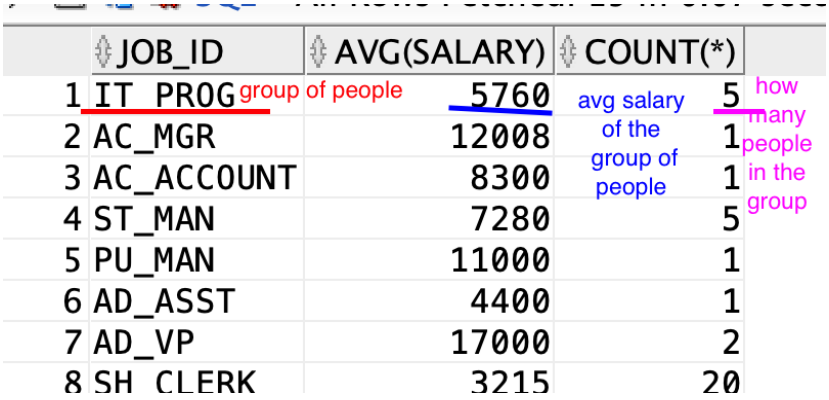
```
--find sum of all salaries
select sum(salary) from employees;
```

GROUP BY

Group by job_id and get avg salary

```
-- get the avg salary for all the kind of jobs
select job_id,avg(salary), count(*),sum(salary)
from employees
group by job_id;
```



	- What is the process inside: It-prog is one box which keep (group) all the employees inf that are working as IT_Prog), count the avg(salary)  group of people avg salary of the group of people how many people in the group
HAVING / WHERE - Group by with condition	1. Eliminate the people **before** putting them into group 2. **After** we created the group
Group by example	Show all job_id and average salary who work as any of these jobs IT_PROG, SA_REP, FI_ACCOUNT, AD_VP

JOB_ID	AVG(SALARY)	COUNT(*)
1 IT_PROG	5760	5
2 AC_MGR	12008	1
3 AC_ACCOUNT	8300	1
4 ST_MAN	7280	5
5 PU_MAN	11000	1
6 AD_ASST	4400	1
7 AD_VP	17000	2
8 SH_CLERK	3215	20

```
--get me job_ids where their avg salary is more than 5k before
--group the people into group
select job_id,avg(salary), count(*),sum(salary)
from employees
where salary >5000
group by job_id;
```

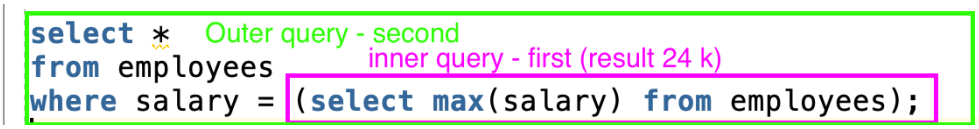
```
--get me job_ids where their avg salary is more than 5k after
-- you will place people into group
select job_id,avg(salary), count(*),sum(salary)
from employees
group by job_id
having avg(salary)>5000;
```

```
select JOB_ID, avg(salary) from EMPLOYEES
where JOB_ID in ('IT_PROG','SA_REP','FI_ACCOUNT', 'AD_VP') group by JOB_ID ;
```

The having statement sets the condition for group rows created by the **GROUP BY** clause
after the GROUP BY applies.

WHERE clause sets the condition for individual rows **BEFORE GROUP BY** clause applies.

QUERY/SUBQUERY

<table>
<tr><td>Subquery</td><td>- inner query always will be executed first, next outer query will be executed.
How to find a person who is making the highest salary in the company - dynamically (no hardcoding):
 


```
select *
from employees
where salary = (select max(salary) from employees);
```


We could do it in two steps static - not practical

```
--how to find employees information who is making high slar
select * from employees
order by salary desc;

--get me higher salary
select * from employees
WHERE salary = 24000;
```

</td></tr>
<tr><td colspan="2">IQ get me the information of the person who is getting the second highest salary

```
-- find the employee inf who is getting second higest salary
select * from employees
where salary= (select max(salary) from employees where salary 2
< (select max(salary) from employees)); 1
```

WE COMBINED 3 QUERY TOGETHER 3

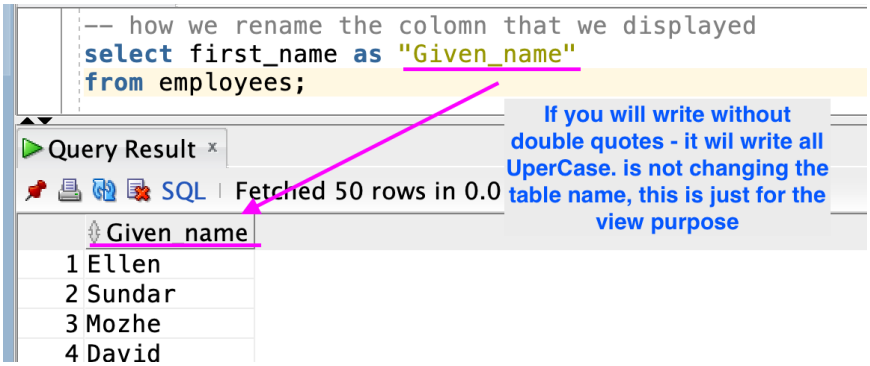
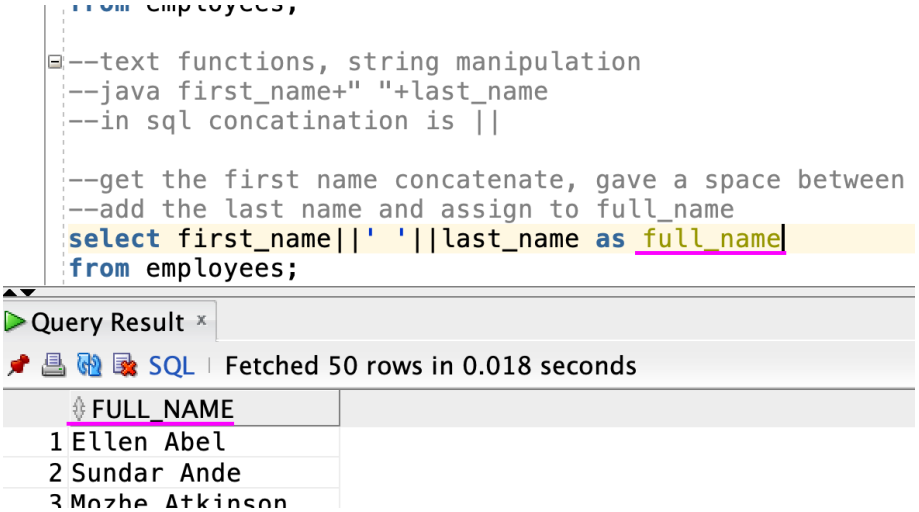
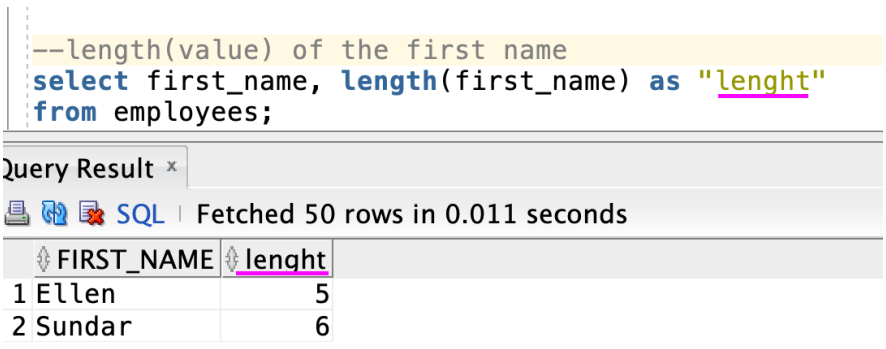
EXAMPLE COMBINES 3 QUERY TOGETHER</td></tr>
<tr><td>Do order by asc, desc</td><td>

```
--order all empl based on salary high to low then display fisrt 10 result
select * from (select * from employees order by salary desc)
where rownum < 11;
```

Giving order by desc</td></tr>
<tr><td>rownum</td><td>Allows you to sort after the result of any condition that you need in your query.
Limits the number of results displayed in the query.

```
WHERE rownum <11
```


If we want to order table first based on our needs(salary high to low) then use query as a table to get number of rows</td></tr>
</table>

how we rename the column that we displayed	
Concatenation in the SQL	
Make LOWER_CASE	Make the new column with full email and make it lowercase (same to upper), just word upper `--add @gmail.com and name new column to full_email` `select lower(email ||'@gmail.com') as "full_email"` `from employees;`
Length	

```
--length(value) of the first name
select first_name, length(first_name) as "lenght"
from employees
order by "lenght" desc;  Desc order
```

SubString
substr

```
--substr(colName, begIndex, NumberOfChar)
select substr(first_name,0,1)||'-'
||substr(last_name,0,1) as "intials"
from employees;
```

Combination of the 3 Concatenated requirements

```
select substr(first_name,0,1)||'-'
||substr(last_name,0,1) as "intials",
first_name||' '||last_name as full_name,
lower(email ||'@gmail.com') as "full_email"
from employees;
```

Query Result

SQL | Fetched 50 rows in 0.025 seconds

	intials	FULL_NAME	full_email
1	E-A	Ellen Abel	eabel@gmail.com
2	S-A	Sundar Ande	sande@gmail.com

VIEW

Virtual tables. View does not contains the query

```
--VIEW - Virtual Table
CREATE VIEW Emaillist as select substr(first_name,0,1)||'-'
||substr(last_name,0,1) as "intials",
first_name||' '||last_name as "full_name",
lower(email ||'@gmail.com') as "full_email"
from employees;

select full_name from Emaillist;
```

Script Output | Query Result

SQL | Fetched 50 rows in 0.013 seconds

	FULL_NAME
1	Ellen Abel

Get min salary from 14 highest salary

```
--find the highest 14th salary - no duplicates
select min(salary)
from (select DISTINCT salary from employees order by salary desc)
where rownum <15;
```

IQ find employee info who is making 14th highest salary without duplicates

```
--find employee info who is making 14th highest salary
--without duplicates
select first_name, salary
from employees
where salary =
(select min(salary)
from (select distinct salary from employees order by salary desc)
where rownum <15);
```

STEPS

1. List salary high to low (desc) without duplicates - *order by salary desc*
2. Cut my list after 14 high salary - *where rownum <15*
3. Get the lowest from the 14th highest salary - *select min(salary)*
4. Now I need to find the employees in who is making lowest salary from 14th low salary → I will create new query - *select first_name, last_name from employee where salary (ADD THE PREVIOUS 3 STEPS(PREVIOUS QUERY THAT FOUNDED LOWEST SALARY FROM 14TH))*

CREATE TABLE/JOIN

DDL: Data Definition Language - used to define data structures:

- CREATE
- DROP
- TRUNCATE
- ALTER

DML: Data Manipulation language - used to manipulate data itself (most likely we will not work with this, but we will learn it):

- SELECT
- INSERT
- UPDATE
- DELETE

Create table:

- Create a new table SQL, you use the CREATE TABLE statement
- First you specify the name of the new table after the create table clause.
- CONSTRAINT (LIMITATION) IS NOT MANDATORY

COLUMN CONSTRAINTS(limitations/some rules):

- NOT NULL - the value of the column cannot be NULL
- UNIQUE - value must be unique across the whole table
- PRIMARY KEY - Combination of both NOT NULL and UNIQUE constraints
- REFERENCES - another table (PKColumn) - used to give foreign keys to the column. We give the foreign key to the column by making Other table primary key column as the reference column

In order to have foreign key, we must have the primary key in other table

<table>
<tr><td></td><td>

```
Create table ScrumTeam(
Emp_ID Integer primary key,
FirstName varchar(30) not null,
LastName varchar(30),
JobTitle varchar(20)
);
```

</td></tr>
<tr><td>varchar</td><td>

```
LastName varchar(30)
```

Varchar - Data type for accepting any character.
30 - is limitation</td></tr>
<tr><td>INSERT</td><td>

- Value list must be in the same order as you have in the table

```
INSERT INTO tableName (column1, column2,...)
VALUES (value1, value2 ... );
```

</td></tr>
<tr><td>UPDATE</td><td>

```
UPDATE table_name
SET column1 = value1,
    column2 = value2 , ...
WHERE condition;
```

</td></tr>
</table>

	`update newteam set salary = 120000 where emp_id = 2;`
DELETE	`Delete from NewTeam` `where emp_id = 1;`
COMMIT	Save the information `commit;`
ALTER	ADD COLUMN `select * from NewTeam;` `--adding new column` `alter table newteam add salary Integer;` Name of the new Column `--rename column` `alter table newteam RENAME column salary to annual_salary;` `--remove(delete) the column` `alter table newteam DROP COLUMN annual_salary;` `--TRUNCATE, if we want to delete all data but still want to` `-- keep the table structure, we use TRUNCATE` `-- delete the data, but still keep the table` `TRUNCATE TABLE agileteam;`

```
--how to change table name?
alter table newteam RENAME TO agileteam;

select * from agileteam;
```

CREATE new SELECT

Script Output | Query Result

SQL | Executing:select * from NewTeam in 0 seconds

```
ORA-00942: table or view does not exist
00942. 00000 -  "table or view does not exist"
*Cause:
*Action:
Error at Line: 1 Column: 15
```

When you change the name and you RUN with old SELECT it will give you ERROR.

```
--If you want to delete the table and data together
DROP TABLE agileteam;
```

Script Output | Query Result

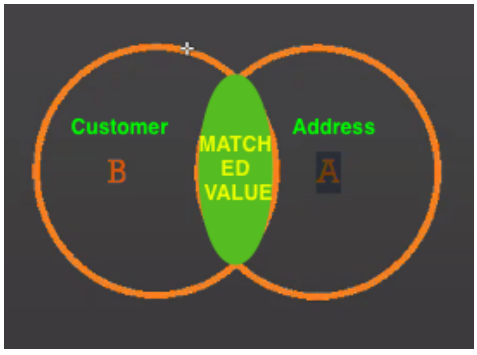
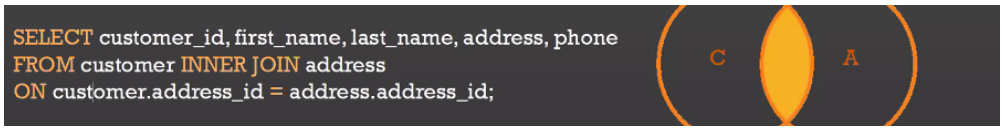
SQL | Executing:select * from agileteam in 0 seconds

```
ORA-00942: table or view does not exist
00942. 00000 -  "table or view does not exist"
*Cause:
*Action:
Error at Line: 18 Column: 15
```

If you DROPED the table , and run old SELECT - ERROR

- The popular actions that we can do with alter keyword:
  - ADD COLUMN : adds column to the table
  - DROP COLUMN : drops the column from the table
  - RENAME COLUMN : renames the column name
  - RENAME TO : renames the table name

JOIN	In order to get F, L, phone num from first table, by comparing the address id. In Customer table you have foreign key for address_id. In Address table primary key address_id `select first_name, last_name, address, phone` `from customer join address on customer.address_id = address.address_id;` I want to get information from two tables and put them in one new table.

	They are connected by the ADDRESS_ID Customer c_id | first_name | last_name | address_id 1 | Mary | Smith | 5 2 | Patricia | Johnson | NULL 3 | Linda | Williams | 7 4 | Barbara | Jones | 8 5 | Elizabeth | Brown | NULL Address address_id | address | phone 5 | 1913 Hanoi Way | 28303384 7 | 692 Joliet Street | 44847719 8 | 1566 Inegl Manor | 70581400 10 | 1795 Santiago | 86045262 11 | 900 Santiago | 16571220 first_name | last_name | address | phone Mary | Smith | 1913 Hanoi Way | 28303384 - Customer has FOREIGN key - for the ADDRESS_ID - Address has PRIMARY key - from the ADDRESS_ID - ON is the KEY word `on customer.address_id = address.address_id;` THIS IS TEMPORARY, WE ARE NOT CHANGING THE NAME OF THE TABLE ``` select first_name, last_name, address, phone from customer "CUST" join address a on CUST.address_id = a.address_id; ``` We are making custom name, in order not type whole class name - IS NOT CHANGING THE TABLE NAME - TEMPORARY
INNER JOIN	INNER JOIN:If the value that is matching from both side, it will get the value from the INNER join INNER JOIN - produce only the set of records that match in both Table Customer and Table Address  
LEFT JOIN	LEFT OUTER JOIN -

	When we have records that are not match it will still take it but → it will create the value by default - NULL - left is what after FROM and BEFORE the JOIN Word `select first_name,last_name,c.address_id,a.address,phone` `from customer c left join address a` `on c.address_id = a.address_id;` Query Result SQL | All Rows Fetched: 5 in 0.047 seconds |	FIRST_NAME	LAST_NAME	ADDRESS_ID	ADDRESS	PHONE | |---|---|---|---|---|---| | 1 | Mary | Smith | 5 | 1913 Hanoi Way | 28303384 | | 2 | Patricia | Johnson | (null) | (null) | (null) | | 3 | Linda | Williams | 7 | 692 Joliet Street | 44847719 | | 4 | Barbara | Jones | 8 | 1566 Inegl Manor | 70581400 | | 5 | Elizabeth | Brown | (null) | (null) | (null) | 
RIGHT JOIN	RIGHT OUTER JOIN - After JOIN When we have records that are not match it will still take it but → it will create the value by default - NULL


```
select first_name,last_name,c.address_id,a.address,phone
from customer c right join address a
on c.address_id = a.address_id;
```

Query Result

SQL | All Rows Fetched: 5 in 0.02 seconds

	FIRST_NAME	LAST_NAME	ADDRESS_ID	ADDRESS	PHONE
1	Mary	Smith	5	1913 Hanoi Way	28303384
2	Linda	Williams	7	692 Joliet Street	44847719
3	Barbara	Jones	8	1566 Inegl Manor	70581400
4	(null)	(null)	(null)	900 Santiago	16571220
5	(null)	(null)	(null)	1795 Santiago	86045262

FULL JOIN

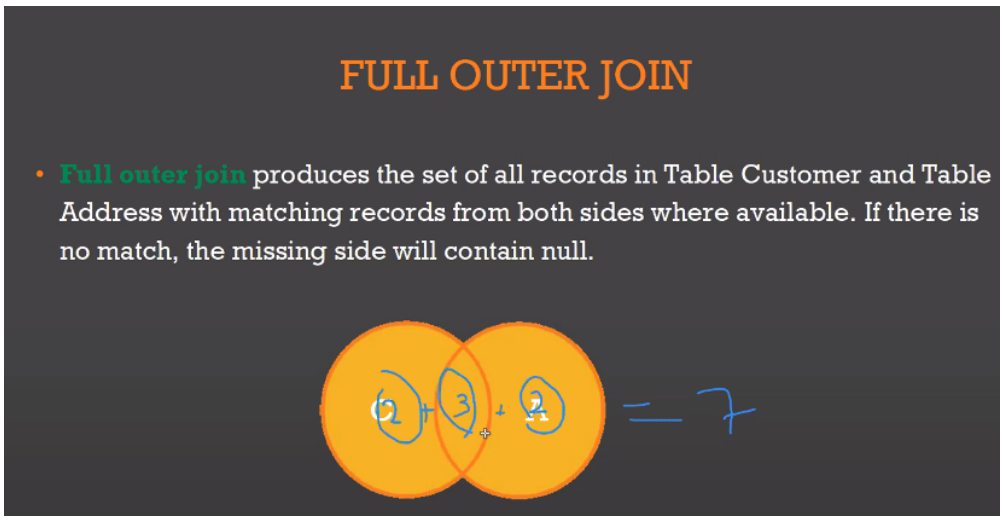
FULL JOIN - will show the information for the RIGHT, LEFT side

```
select first_name,last_name,c.address_id,a.address,phone
from customer c full join address a
on c.address_id = a.address_id;
```

Query Result

SQL | All Rows Fetched: 7 in 0.019 seconds

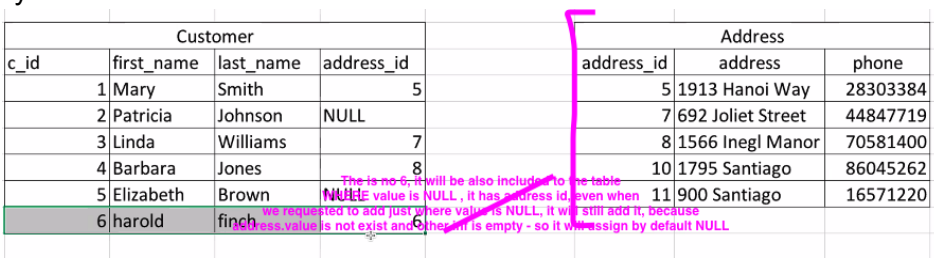
	FIRST_NAME	LAST_NAME	ADDRESS_ID	ADDRESS	PHONE
1	Mary	Smith	5	1913 Hanoi Way	28303384
2	Linda	Williams	7	692 Joliet Street	44847719
3	Barbara	Jones	8	1566 Inegl Manor	70581400
4	(null)	(null)	(null)	1795 Santiago	86045262
5	(null)	(null)	(null)	900 Santiago	16571220
6	Elizabeth	Brown	(null)	(null)	(null)
7	Patricia	Johnson	(null)	(null)	(null)



JOIN/SELF JOIN/SET OPERATORS

JOINS	**INNER JOIN** Inner join (definition) intersections between the tables. Matching information from the both Tables If you only use a join key word by default is - INNER JOIN
	Left - you wanna have everything from the left table (LEFT OUTER or LEFT - KEY WORD) Left join with WHERE - to produce the set of records only in Customer Table, but not in Address Table. WHERE KEY WORD - giving specification to what you want to have in the table. ***(I want rows WHERE ID value is 5)*** *WE ARE JUST INTERESTED TO GET INFORMATION FOR THE LEFT SIDE PEOPLE WHO IS **VALUE IS NULL*** `Select customer_id, first_name, last_name, c` `address.address_id address, phone` `from customer Left OUTER JOIN address` `on customer.address_id = address.address_id` `where customer.address_id is NULL;`
	There is no 6 value, it will also include the table WHERE value is NULL, it has address id, but it is missing other information, and also address_id does not exist in Address table. Therefore even when we requested to add just value which is

address_id NULL, it will still add it, because address value is not exist so by default it will be added



Customer

c_id	first_name	last_name	address_id
1	Mary	Smith	5
2	Patricia	Johnson	NULL
3	Linda	Williams	7
4	Barbara	Jones	8
5	Elizabeth	Brown	NULL
6	harold	finch	6

Address

address_id	address	phone
5	1913 Hanoi Way	28303384
7	692 Joliet Street	44847719
8	1566 Inegl Manor	70581400
10	1795 Santiago	86045262
11	900 Santiago	16571220

If the case, when you do want to see everything from both table, but you are puting flexible condition

WHERE customer.address_id is NULL or address.address_id is NULL

```
Select *
from customer Full OUTER JOIN address
on customer.address_id = address.address_id
where customer.address_id is NULL OR
address.address_id is NULL;
```

JOIN ONE 3 AND MORE TABLES

```
--get first, last and dep_name, city, country_name for all empl
select first_name,last_name, department_name, city, country_name
from employees e join departments d
on e.department_id=d.department_id
join locations l
on d.location_id = l.location_id
join countries c
on l.country_id = c.country_id;
```

We have 3 INNER JOIN, and 3 connections ON

SELF JOIN

SELF JOIN - JOIN THE SAME TABLE

Relations between two tables

If employees table Manager ID has same value in the Employees ID, get the First name and Last Name of the Manager

```
select e1.employee_id,e1.first_name,e1.last_name,
e1.manager_id,e2.employee_id,e2.first_name,e2.last_name
from employees e1 left join employees e2
on e1.manager_id = e2.employee_id
order by e1.employee_id;
```

	EMPLOYEES emp_id | first_name | last_name | manager_id | emp_id | manager_f_n | manager_l_n 100 | Steven | King 101 | Neena | Kochhar | 100 | 100 | Steven | King 102 | Lex | De Haan | 100 | 100 | Steven | King emp_id | first_name | last_name | manager_id 100 | Steven | King 101 | Neena | Kochhar | 100 102 | Lex | De Haan | 100 103 | Alexander | Hunold | 102 Steven is ID == To Manager ID Steven is manager Nenna and Lex. Steven does not have manager
SET OPERATORS	- You need 2 independent queries - Same number of columns in the select statement - Same data type in the same order There is not connections between table no foreign key How are we gonna connect? UNION - operator combines result sets of two or more SELECT statements. The ***UNION*** operator removes all duplicates rows unless the ***UNION ALL*** ***UNION*** - sorts the results directly if there is duplicate it will be removed IN THIS EXAMPLE - STEVEN IS HAVING DIFFERENT SALARY AND ID - SO IS NOT COUNT AS DUPLICATES, BUT IF YOU WILL ONLY CALL NAME WITHOUT SALARY AND ID IT WILL BE REMOVED 1 select * from Testers 2 union 3 select * from Developers; Query Result SQL | All Rows Fetched: 7 in (ID_NUMBER | NAMES | SALARY 1 | 1 | Mike | 155000 2 | 1 | Steven | 110000 3 | 2 | Adam | 105000 4 | 2 | John | 142000 5 | 3 | Lex | 100000 6 | 3 | Steven | 850000 7 | 4 | Maria | 120000 1 select testers.names from Testers 2 union 3 select developers.names from Developers; Query Result SQL | All Rows Fetched: 6 in 0.01 seconds NAMES 1 Adam 2 John 3 Lex 4 Maria 5 Mike 6 Steven ***UNION ALL*** - will not sort, will not delete duplicates - will keep ALL

```
1 select * from Testers
2 union all
3 select * from Developers;
```

Query Result | SQL | All Rows Fetched: 7 in 0.0

	ID_NUMBER	NAMES	SALARY
1	1	Steven	110000
2	2	Adam	105000
3	3	Lex	100000
4	1	Mike	155000
5	2	John	142000
6	3	Steven	850000
7	4	Maria	120000

MINUS

MINUS - it will only get unique query FROM THE FIRST SELECT and minus the duplicates in the table

- Returns records from the first that is not present in the second, it will only return values that are not coming in

TESTER FIRST - Second DEVELOPERS

```
select names from Testers
minus
select names from Developers;
```

SQL | All Rows Fetched: 2 in 0.014 se

	NAMES
1	Adam
2	Lex

DEVELOPERS FIRST - Second TESTERS

```
select names from Developers
minus
select names from Testers;
```

SQL | All Rows Fetched: 3 in 0.01

	NAMES
1	John
2	Maria
3	Mike

STEVEN is still in the query, because the ID and SALARY is UNIQUE

```
select * from Testers
minus
select * from Developers;
```

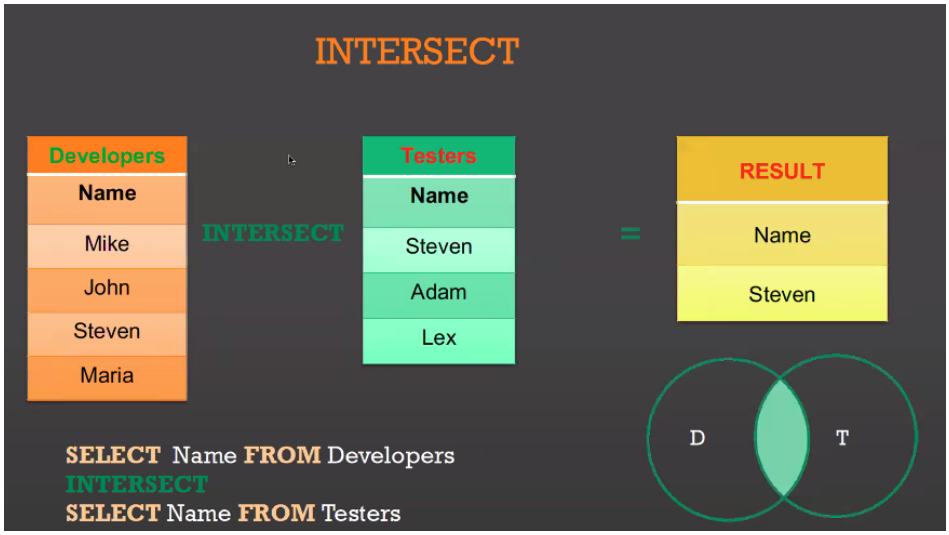
SQL | All Rows Fetched: 3 in 0

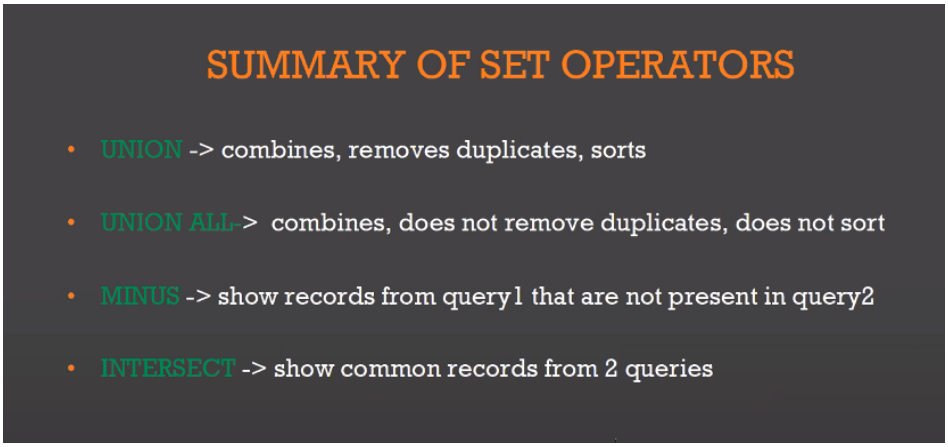
	ID_NUMBER	NAMES	SALARY
1	1	Steven	110000
2	2	Adam	105000
3	3	Lex	100000

```
select * from Developers
minus
select * from Testers;
```

SQL | All Rows Fetched: 4 in (

	ID_NUMBER	NAMES	SALARY
1	1	Mike	155000
2	2	John	142000
3	3	Steven	850000
4	4	Maria	120000

INTERSECT	Will keep what is common, the only common thing is name, so if we will call ID and SALARY it will not work `select names from Testers` `INTERSECT` `select names from Developers;` uery Result SQL | All Rows Fetched: 1 in 0.012 NAMES 1 Steven 



IQ - How to find duplicate names in employees table

- NO - DISTINCT or GROUP BY - because it will remove the duplicates

We need:

1. Select just First names

2. Count of each first name
3. Group by first name
4. Having count(*)>1

```
select first_name, count(*)
from employees
group by first_name
having count (*)>1;
```

ery Result

SQL | All Rows Fetched: 13 in

FIRST_NAME	COUNT(*)
Peter	3
Michael	2
Steven	2
John	3
Julia	2
William	2
Karen	2
Kevin	2
David	3
Jennifer	2
Randall	2
Alexander	2
James	2

```
select first_name, count(*)
from employees
group by first_name;
```

ery Result

SQL | Fetched 50 rows in 0.05 s

FIRST_NAME	COUNT(*)
Peter	3
Michael	2
Shelley	1
Steven	2
Samuel	1
Christopher	1
Lindsey	1
Sigal	1

Statement:

= → equal
>/< → greater than/ less then
>=/<= → greater.equal/less.equal
<> or != → not equal
AND → logical AND (both has to be), unless its working with keyword *between*
OR → logical OR (one or another)

SQL:

- Case NOT Sensitive
- Statements end at the semicolon, you can go to the second line…

WE WILL NOT WORK WITH NON-RELATION-DATABASE