SQL
For QA Engineers

PART 4

SELECT…WHERE (JOIN)

EMPLOYEES

Last_Name	First_Name	Department_Id
Patel	Anu	15
Beyer	**David**	**80**
Chen	**Alice**	**80**
Johnson	**John**	**80**
Pham	**Julia**	**80**
Alison	David	120

DEPARTMENTS

Department_Id	Department_Name
10	Accounting
20	Payroll
80	**IT**

RESULTSET

Last_Name	First_Name	Department_Name
Beyer	**David**	**IT**
Chen	**Alice**	**IT**
Johnson	**John**	**IT**
Pham	**Julia**	**IT**

Read employee and department name from two related tables:

```
SELECT  e.FIRST_NAME,
        e.LAST_NAME,
        d.DEPARTMENT_NAME
FROM    EMPLOYEES   e,
        DEPARTMENTS d
WHERE   d.department_ID = e.department_ID
```

Question

EMPLOYEES

Last_Name	First_Name	Department_Id
Patel	Anu	15
Beyer	**David**	**80**
Chen	**Alice**	**80**
Johnson	**John**	**80**

DEPARTMENTS

Department_Id	Department_Name
20	Payroll
80	**IT**

How would you:

1) Read **ALL employees**, and department names when available; *OR*
2) Read **ALL departments**, with employees, when available; *OR*
3) Read ALL employees with department names, when available and show ALL departments.

Last_Name	First_Name	Department_Name
Patel	**Anu**	
Beyer	David	IT
Chen	Alice	IT
Johnson	John	IT

Last_Name	First_Name	Department_Name
		Payroll
Beyer	David	IT
Chen	Alice	IT
Johnson	John	IT

Last_Name	First_Name	Department_Name
Patel	**Anu**	
Beyer	David	IT
Chen	Alice	IT
Johnson	John	IT
		Payroll

JOIN

(Related Tables)

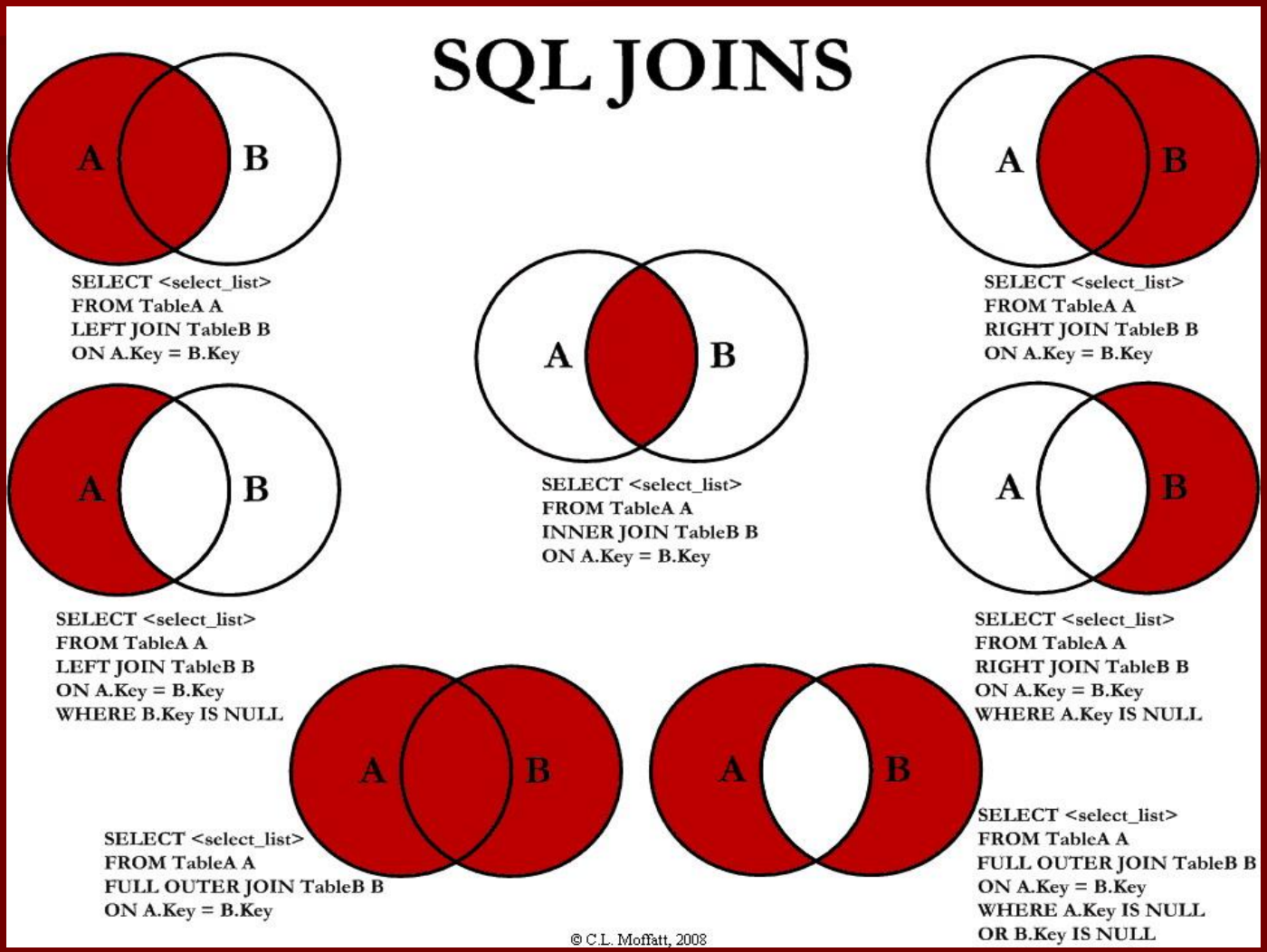
Types of Joins

- **Inner join** (default) - A join that displays only the rows that have a match in both joined tables.
- **Outer join** - A join that includes rows even if they do not have related rows in the joined table. There are 3 types of outer (unmatched rows to be included) joins:
  - **Left outer join**
  - **Right outer join**
  - **Full outer join;**
- **Cross Join** - A join whose result set includes one row for each possible pairing of rows from the two tables.

Popular Interview Questions!

More JOINS (Self Study)

http://en.wikipedia.org/wiki/Join_%28SQL%29



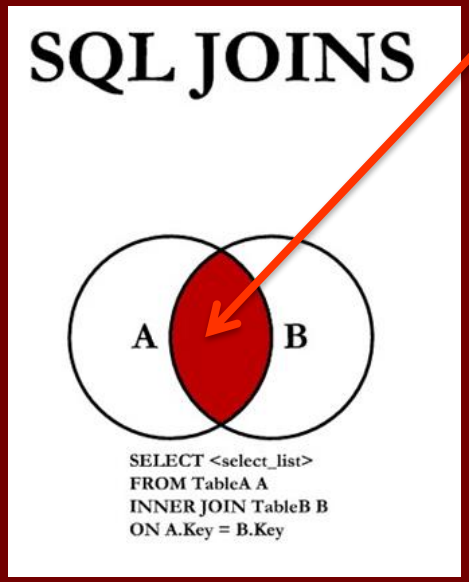
INNER JOIN

An [INNER] JOIN (default) is the most common join. It joins every record in table A with a record in table B, which satisfy the join conditions

```
SELECT        *
FROM          EMPLOYEES
[INNER] JOIN  DEPARTMENTS
ON            EMPLOYEES.Department_ID =
              DEPARTMENTS.Department_ID
```

OR

```
SELECT *
FROM   EMPLOYEES, DEPARTMENTS
WHERE EMPLOYEES.Department_ID = DEPARTMENTS.Department_ID
```



Columns with NULL do not match any other values.

INNER JOIN

To add condition(s) to your join:

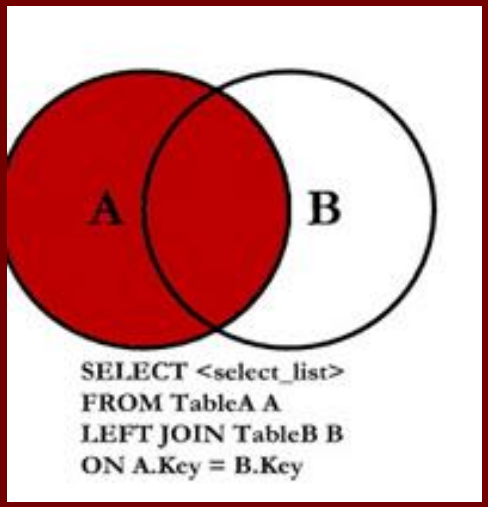
```
SELECT      *
FROM        EMPLOYEES       e
JOIN        DEPARTMENTS     d
ON          e.Department_id = d.Department_id
WHERE       department_id = 80
```

OUTER JOIN

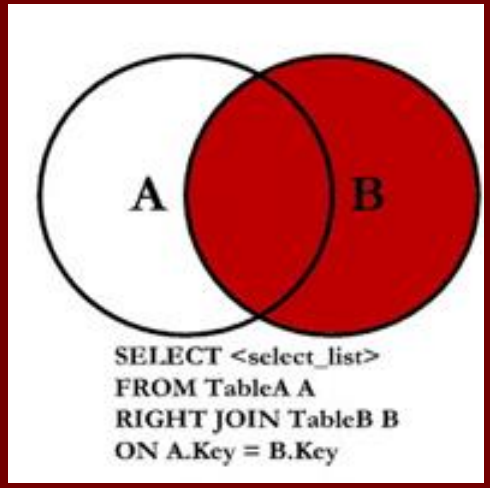
OUTER JOIN does not require each record in the two joined tables to have a matching record.

Depending on which table(s) one retains the rows (left, right, or both), there are:

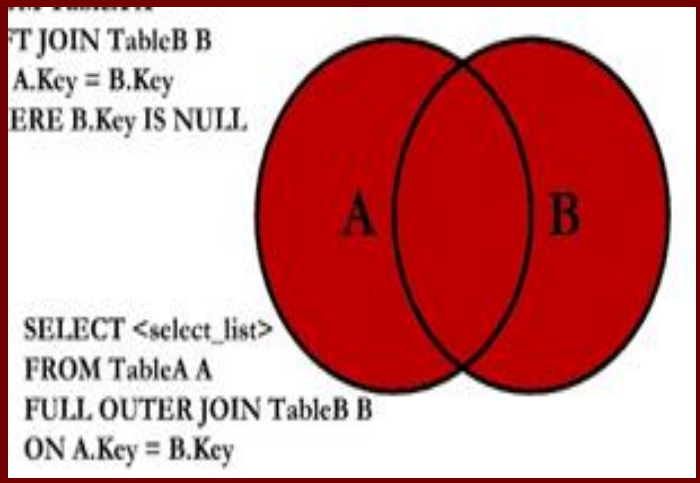
Left outer join



Right outer join



Full outer join



CROSS JOIN

CROSS JOIN will produce rows which combine each row from the first table with each row from the second table.

```
SELECT      *
FROM        EMPLOYEE
CROSS JOIN DEPARTMENT
```

OR

```
SELECT *
FROM   EMPLOYEE, DEPARTMENT
```

It's like select from two tables without a WHERE clause. Return is huge and cross join is not that popular.

Exercise

Read all employees from employee table:

```
select    e.last_name,
          e.first_name,
          e.department_id
from      employees  e
```

Total 107 employees on my screen

Exercise

Read employees with the matching department names:

```
select    e.last_name,
          e.first_name,
          e.department_id,
          d.department_name
from      employees  e,
          departments d
where     d.department_id = e.department_id
```

Total 106 employees on my screen
*We can rewrite it using **INNER JOIN!***

Exercise

Read employees with the matching department names:

```
select    e.last_name,
          e.first_name,
          e.department_id,
          d.department_name
from      employees  e
join      departments d
on        d.department_id = e.department_id
```

Same 106 employees on my screen
INNER JOIN is the same SELECT…WHERE

Exercise

Why from the total of 107 employees, only 106 are retrieved?

*How would you read **all 107 employees** with matching department name, only when it's available?*

Exercise

Read **ALL** employees with the matching department names, when it's available:

Abel	Ellen	80	Sales
Hutton	Alyssa	80	Sales
Taylor	Jonathon	80	Sales
Livingston	Jack	80	Sales
Grant	Kimberely	-	-
Johnson	Charles	80	Sales
Taylor	Winston	50	Shipping
Fleaur	Jean	50	Shipping
Sullivan	Martha	50	Shipping
Geoni	Girard	50	Shipping

```
select    e.last_name,
          e.first_name,
          e.department_id,
          d.department_name
from      employees  e
left join departments d
on        d.department_id = e.department_id
```

107 employees, with one employee, missing the department name

Exercise

*Now, let's try to read **all available departments** with matching employee, when it's available?*

Exercise

Read **all available departments** with matching employee, when it's available :

```
select    e.last_name,
          e.first_name,
          e.department_id,
          d.department_name
from      employees  e
right join departments d
on        d.department_id =
          e.department_id
```

Many retrieved departments do not have employees

Urman	Jose Manuel	100	Finance
Sciarra	Ismael	100	Finance
Higgins	Shelley	110	Accounting
Gietz	William	110	Accounting
-	-	-	Treasury
-	-	-	Corporate Ta
-	-	-	Control And
-	-	-	Shareholder
-	-	-	Benefits
-	-	-	Manufacturir
-	-	-	Construction
-	-	-	Contracting
-	-	-	Operations
-	-	-	IT Support
-	-	-	NOC
-	-	-	IT Helpdesk
-	-	-	Government
-	-	-	Retail Sales
-	-	-	Recruiting
-	-	-	Payroll

127 rows returned in 0.04 seconds CSV Export

Exercise

As a last exercise, let's retrieve all employees with matching departments and all departments, with matching employees (two complete tables)

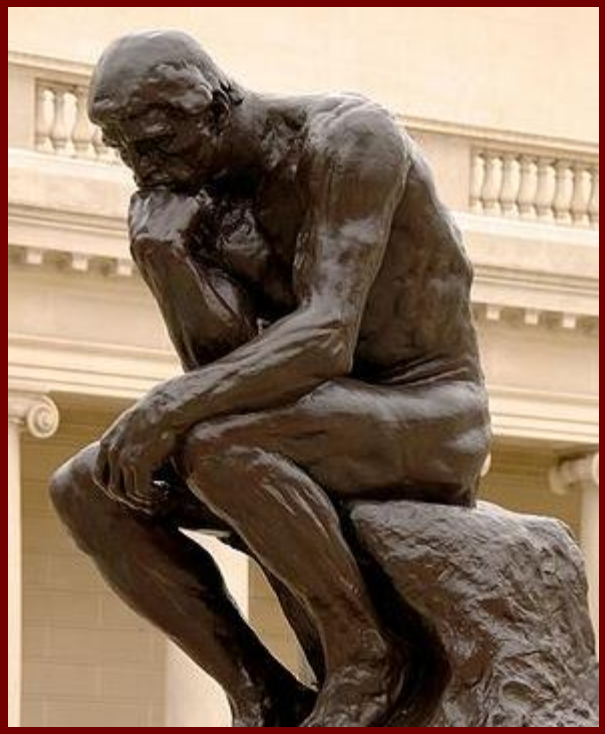
Exercise

Read **all available departments** with matching employee, when it's available :

```
select    e.last_name,
          e.first_name,
          e.department_id,
          d.department_name
from      employees  e
full join departments d
on        d.department_id = e.department_id
```

There are 123 records back!

Quiz



Read ***all employees***,
whose last name starts with ‘G’ and if available, display department name;

Question

EMP_UK

ID	Last_Name	First_Name	Salary
1	Patel	Anu	1500
2	Beyer	David	8000
3	Chen	Alice	2300
4	Johnson	John	3500

EMP_ASIA

ID	SAL	LNAME	FNAME
1001	900	Stevenson	David
1002	120	Pham	Julia

How would you read all company employees & salaries?

UNION

(NOT Related Tables)

UNION

UNION is used to combine the result-set of two or more SELECT statements.

```
SELECT column_name(s) FROM table_name1
UNION
SELECT column_name(s) FROM table_name2
```

- Each SELECT must have the same number of columns.
- Selected column(s) must have similar data types.
- Selected column(s) must be in the same order.

UNION operator selects *only distinct (unique) values* by default.

UNION ALL

May UNION select ALL data, not only distinct?

To allow *duplicate values*, use UNION ALL.

```
SELECT column_name(s) FROM table_name1
UNION ALL
SELECT column_name(s) FROM table_name2
```

UNION

EMP_UK

ID	Last_Name	First_Name	Salary
1	Patel	Anu	1500
2	Beyer	David	8000
3	Chen	Alice	2300
4	Johnson	John	3500

EMP_ASIA

ID	SAL	LNAME	FNAME
1001	900	Stevenson	David
1002	120	Pham	Julia

How would you read all company employees & salaries?

```
Select  last_name, first_name, salary
From    EMP_UK
UNION ALL
Select  lname, fname, sal
From    EMP_ASIA
```

Last_Name	First_Name	Salary
Patel	Anu	1500
Beyer	David	8000
Chen	Alice	2300
Johnson	John	3500
Stevenson	David	900
Pham	Julia	120

Join vs. Union

UNION combines the results of *two or more queries* into a *single result set* that includes *all the rows* that *belong* to *all queries in the union*.

JOINs retrieve data from *two or more tables* based on *logical relationships between the tables*. Joins indicate how *SQL should use data from one table to select the rows in another table*.

UNION is different from JOINs that combine columns from two tables.

Join vs. Union - Example

Emp_Name (a)

LastName	FirstName
Singh	Anjana
Alentova	Katia

Dept_List (b)

Last_Name	First_Name	Country
Patel	Avinash	USA
Stepin	Ivan	USA
Singh	Anjana	Canada

UNION Example:

```
SELECT LastName, FirstName
FROM   Emp_Name
UNION
SELECT Last_Name, First_Name
FROM   Dept_List
```

UNION Output:

Singh	Anjana
Alentova	Katia
Patel	Avinash
Stepin	Ivan

JOIN Example:

```
SELECT  a.LastName, a.FirstName, b.Country
FROM    Emp_Name a
INNER JOIN Dept_List b
ON  a.LastName = b.LastName
```

JOIN Output:

Singh	Anjana	Canada

Commit vs. Roll Back; Comment

Transaction is a logical block of SQL statements that succeed or fail.

COMMIT - permanently update the dB

ROLL BACK - undo your changes (transaction failed, etc)

/*…*/ - Oracle comments

```
SELECT column1 FROM table1   /* Comments Example */
```

More Advanced…

(Optional Self-Study)

VIEW

VIEW is a virtual table which is created by executing a query. It is an abbreviation of SELECT statement.

Example:
At work, I need the list of Bay Area employees all the time. For that every time, I type:

```
SELECT  FIRST_NAME, LAST_NAME
FROM    EMPLOYEES
WHERE   PHONE_NUMBER LIKE '650.%'
```

Or… I can store this statement as a view 'local_employees_v ' (equivalent to the SELECT)

```
SELECT * FROM local_employees_v
```

VIEW Example

Compare two statements, which retrieve all Bay Area stock managers:

```
SELECT   FIRST_NAME, LAST_NAME
FROM     EMPLOYEES
WHERE    PHONE_NUMBER LIKE '650.%'
AND      JOB_ID = 'ST_MAN'
```

```
SELECT FIRST_NAME, LAST_NAME
FROM   local_employees_v
WHERE JOB_ID = 'ST_MAN'
```

How is it different?

Materialized View vs. View

View:

- Does not contain physical data
- Data is retrieved only if a view is executed
- Can be updated with UPDATE statement (NOT a VIEW itself, but table(s) it points to)
- It does not bring performance gain.
- It makes referral to SQL statements more simple.

Materialized View vs. View

Materialized view:

- Does contain physical data
- Data is actually stored in materialized view
- Allows to bring performance gain.
- If table(s) is big and we frequently use only some data, then storing that data separately (materialize view) will help me to execute query faster.

Materialized view shall be periodically synchronized with underlying table since it contains the source data

REFRESH FORCE ON DEMAND

Error Creating View

ORA-01031: insufficient privileges

It means that:

- You, as a user, does not have enough privileges to create materialized view/table.
- DBAdmin did not give you enough privileges

Why?

TRIGGER

What is a trigger?

Well, it's a piece of SQL code that is activated when a certain event happens. A common usage is when a new record is added to a database, this triggers is invoked.

For example, if a new employee joined the company (added to table EMPLOYEES) email is sent to manager, IT Department (to prepare his office space), to payroll department, a to LDAP.

INDEXES

INDEX is an optional structure, associated with tables, created to improve query performance.

Just as the index in a book helps you to quickly find specific information, an Oracle index provides a quick access path to table data.

Index can be created on one or more columns of a table;

Index is automatically maintained and used by Oracle

Subsets of SQL-Types of Commands

DDL (Data Definition Language) statements are used to define the database structure or schema (TABLE, VIEW, INDEX, etc.):

- CREATE - to create objects in the database
- ALTER - alters the structure of the database
- DROP - delete objects from the database
- TRUNCATE - remove all records from a table
- COMMENT - add comments to the data dictionary
- RENAME - rename an object

DML (Data Manipulation Language) statements are used for managing data within schema:

- SELECT - retrieve data from the a database
- INSERT - insert data into a table
- UPDATE - updates existing data within a table
- DELETE - deletes all or specific (WHERE clause) records from a table

DCL (Data Control Language) statements:

- GRANT - gives user's access privileges to database
- REVOKE - withdraw access privileges given with the GRANT command

INTERVIEW QUESTION!

INTERVIEW QUESTIONS

For QA Engineers

SQL-GENERAL

Q. How did you use SQL in your project?

Q. Your experience with SQL

Q. From your previous experience that you had with SQL, can you scale from 1-10?

Q. What does SQL stand for?

A. Structured Query Language

Q. What is Normalization?

A. The process of table design is called normalization to minimize data dependency and redundancy.

SQL – DATA TYPES

Q. Describe how NULLs work in SQL?

NULL is how SQL handles missing (UNDEFINED) values. IS NULL or IS NOT NULL are arithmetic commands. It will return a NULL.

Q. List all the possible values that can be stored in a BOOLEAN data field.

A. There are only two values that can be stored in a BOOLEAN data field: -1(true) and 0(false).

Q. What is the highest value that can be stored in a BYTE data field?

A. The highest value that can be stored in a BYTE field is 255. or from -128 to 127.

Q. How user defined data types and when you should go for them?

A. Define data types when creating or altering table. Then you need to specify column and datatype. For example:

```
CREATE TABLE table1 (ID int, Last_Name varchar(255))
ALTER TABLE table1 ADD column1 varchar(255)
ALTER TABLE table1 MODIFY column1 varchar(255)
```

SQL – PK/FK

Q. What is a primary key?

Primary key is a unique column(s) in the table

Q. What is the main role of a primary key in a table?

A. The main role of a primary key in a data table is to maintain the internal integrity of a data table.

Q. What are foreign keys?

A. Foreign key is a field that links one table to another table's primary or foreign key. Usually it's not unique

Q. Can a table have more than one foreign key defined?

A. Table can have any number of foreign keys defined. It can have only one primary key defined.

Q. What are constraints? Explain different types of constraints.

A. CONSTRAINT restricts the values in dB)

- Not Null constraint - dB value can't be null
- Unique constraint - multiple rows can't have same value in the same column
- Primary Key constraint - combines a NOT NULL and Unique constraints
- Foreign Key constraint

SQL – DML/DDL/DCL

Q. What is DML, DDL, DCL?

DML and DDL are subsets of SQL.

DDL (Data Definition Language) statements are defining dB schema. DDL commands are CREATE, ALTER, DROP, RENAME, TRUNCATE, for TABLES, INDEXES, VIEWS, etc.

DML (Data Manipulation Language) statements are managing data within schema. DML commands are SELECT, INSERT, UPDATE, DELETE

DCL (Data Control Language) statements: GRANT, REVOKE

Q. Which of the following statements are Data Manipulation Language commands (DML)?

A.	INSERT	DML (Data Manipulation Language)
B.	UPDATE	DML (Data Manipulation Language)
C.	TRUNCATE	DDL (Data Definition Language)
D.	CREATE	DDL (Data Definition Language)
E.	GRANT	DCL (Data Control Language)

A. A and B. The INSERT and UPDATE statements are Data Manipulation Language (DML) commands.

SQL-DELETE/UPDATE

Q. Difference between TRUNCATE, DELETE and DROP commands?

A. DELETE is used to remove some or all (no WHERE clause) rows from a table;

TRUNCATE removes ALL rows from a table and cannot be rolled back;

DROP removes a table from the database, the tables' rows, indexes and privileges;

Q. SQL Update table query. How could you update a cell in SQL table?

A.

```
UPDATE    table1
SET       column1 = 'Smith', column2 = 200
WHERE     column3 = 'ABC'
```

Q. How to solve problem with UPDATE one cell in SQL and not change all the parameters in a column?

A. Add WHERE clause to specify your conditions and set the appropriate value:

```
UPDATE    table1
SET       column1 = 'Smith'    <---- new value
WHERE     column2 = 'ABC'      <---- specified conditions
```

That would not change column parameters. ALTER can change the column parameters (length, datatype, add and delete column, etc);

SQL-SELECT/WHERE

Q. Is the WHERE clause must always appear before the GROUP BY clause in SQL SELECT ?

A. Yes. Only the SELECT and FROM clause are mandatory. The proper order for SQL SELECT clauses is:

```
SELECT ….
FROM     ….
WHERE    ….
GROUP BY….
HAVING ….
ORDER BY….
```

Q. Is the WHERE clause must ALWAYS appear in SQL SELECT ?

A. No. Only the SELECT and FROM clause are mandatory. WHERE clause is used to specify the certain condition, not select the whole table.

Q. When do you use WHERE clause and when do you use HAVING clause?

A. WHERE doesn't work for aggregate functions (avg,count, sum, max, etc). We use HAVING clause instead;

```
SELECT     DEPARTMENT_ID, max(SALARY)
FROM       EMPLOYEES
GROUP BY DEPARTMENT_ID
HAVING    MAX(SALARY) > 10000
```

SQL-General

Q. What is difference between Rename and Alias?

A. When you RENAME table or column (ALTER) you actually changing the name, when you use alias, you are just giving a table or column a "nickname"

Q. Describe SQL comments.

A. SQL comments are introduced by two consecutive hyphens (--) and ended by the end of the line.

Q. What does COMMIT do? ROLL BACK?

COMMIT save all changes made by DML statements to the dB, ROLL BACK will undo all changes

Q. What is the system function to get the current user's user id?

A. SELECT USER FROM DUAL

Q. How do you add record to a table?

A. INSERT into table_name VALUES ('ALEX', 33 , 'M')

Q. How do you add a column to a table?

A. ALTER TABLE Department ADD (AGE, NUMBER);

Write SQL

Q. How do you change value of the field?

A.
```
UPDATE   table_name
SET      field1 = 200,
         field2 = 'ABD',
         field3 = to_date ('2006-03-04 09:29', 'yyyy-mm-dd')
WHERE    field = 'CD'    (CONDITION)
```

Q. How do you find the **number of rows** in a Table?

A. SELECT **count(*)** FROM table_name

Q. How do you select **all** records from the table?

A. SELECT * FROM table_name

Q. Write a SQL SELECT query that returns each city **only once** from Students table? Do you need to order this list with an ORDER BY clause?

A. SELECT **DISTINCT** City FROM Students;

Yes, you need an ORDER BY ASC or DESC, if you need to sort more than one returned CITY

Write SQL

Q: Your database contains customers and their addresses. What SQL command you will use to retrieve all addresses within a specific city?

A. SELECT last_name, first_name, address
FROM EMPLOYEES
WHERE EMPLOYEES.CITY = 'San Francisco'

Q. Write SQL SELECT query that returns the first and last name of each instructor, Salary, and gives **each of them a number**.

A. SELECT FirstName, LastName, Salary, **ROWNUM**
FROM Instructors;

Q. Can one select a **random collection** of rows from a table?

A. Yes. Using SAMPLE clause.
Example:
SELECT * FROM EMPLOYEES **SAMPLE(10);**
10% of rows selected randomly will be returned

Write SQL

Q: There is a % sign in one field of a column. What will be the query to find it?

A.
```
SELECT  column1
FROM    table_name
WHERE  column1 LIKE '%[%]%'
```

Q. Choose all '5' from the field.

A.
```
SELECT  column1
FROM    table_name
WHERE  column1 LIKE '%5%'
```

Q. You issue the following query:

```
SELECT  First_Name FROM EMPLOYEES
WHERE  First_Name LIKE '_A%'
```

Which names would be returned by this query? Choose all that apply.

A. Allen

B. CLARK

C. JACKSON

D. David

A. A and C. '_A% is one missing character (_) and any character(s) after A (%)

SQL-TRIGGER/INDEX/VIEW

Q. What is Trigger?

A. TRIGGER defines a set of actions that are performed in response to an insert, update, or delete operation on a specified table. When such an SQL operation is executed, in this case the trigger has been activated.

Q. What is Index? (Advanced: What are the types of indexes, etc.?)

A. INDEX is an optional structure, associated with tables, created to improve query performance.

Q. What is a difference between View and Materialized View?

A. **VIEW** is a **virtual table** which is created by executing a query. It is an abbreviation of SELECT statement (tableA_v)

MATERIALIZED VIEW – contain **physical data.** For big tables, frequently use only some data. Helps execute queries faster;

SQL-JOIN

Q. What is a join?

A. Join is a process of retrieve pieces of data from different sets (tables) and returns them to the user or program as one "joined" collection of data.

Q. How do you extract information from two tables using SQL?

A. You can use SELECT…WHERE table1.id = table2.id or use JOIN

Q. What is INNER JOIN?

A. Inner Join displays only rows that matched in both joined tables. It's a default and most commonly used join. It is a different syntax for select * from table1, table2 where table1.id=table.id

```
SELECT          *
FROM            EMPLOYEES
[INNER] JOIN    DEPARTMENTS
ON EMPLOYEES.Department_ID = DEPARTMENTS. Department_ID
```

Q. What is OUTER JOIN?

A. Outer join includes rows even if they don't have related rows in joined table.

3 types of outer join are: **Left** outer join, **Right** outer join, **Full** outer join

Q. What is CROSS_JOIN?

A. Cross Join includes one row for each possible pairing of rows from two tables

SQL-JOIN

Q. What is a self join? Explain it with an example.

A. Self-join is a query in which a table is joined to itself. Example: Employees table which contains rows for normal employees as well as managers. So, to find out the managers of all the employees, you need a self join.

```
SELECT   e.Last_Name, m.Last_Name
FROM     EMPLOYEES e, EMPLOYEES m
WHERE    e.ID = m.MGRID
```

Q. Write query to get unique zipcodes from a table which has state, country and zipcodes.

A. SELECT DISTINCT ZIPCODE FROM ADDRESS

Q. Write query to get the unique zip codes from 2 tables.
Table "A" has SSN, State, Zipcodes, and Table "B" has Zipcode, City,

A.
```
SELECT DISTINCT ZIPCODE FROM TABLEA
UNION
SELECT DISTINCT ZIPCODE FROM TABLEB
```

Q. Table with state and zipcode columns. Write a query to get unique zipcodes from each state?

```
SELECT DISTINCT state, zipcode
  FROM LOCATION
```

Q: There is a % sign in one field of a column. What will be the query to find it?

A. SELECT column1 FROM table_name
WHERE column1 LIKE '%**[%]**%'

Q. Which SQL statement would you use to extract data from a database?

A. SELECT

Q. How can you change "12-03-1967" to "12-23-1967" in DOB (date of birth) column in Employees table?

A. UPDATE Employees SET DOB = '12-23-1967' WHERE DOB = '12-03-1967'
The whole column DOB will be updated without WHERE clause

Q. Which statement would you use to delete data from a database?

A. DELETE FROM City
WHERE city = 'San Franmcisco'

Q. Please select all records from table "Employees" where value of the column "LastName" contains "ba";

A. SELECT * FROM EMPLOYEES WHERE LastName LIKE '%ba%'

Q. How would you sort the results?

A. SELECT * FROM EMPLOYEES
ORDER BY LastName ASC (default a-z), DESC (z-a)