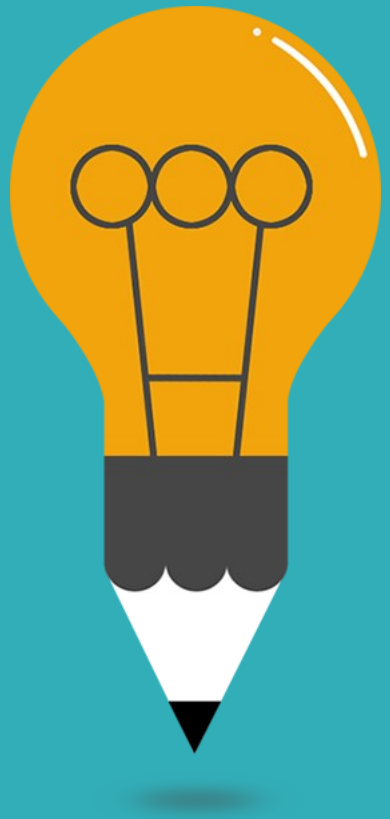


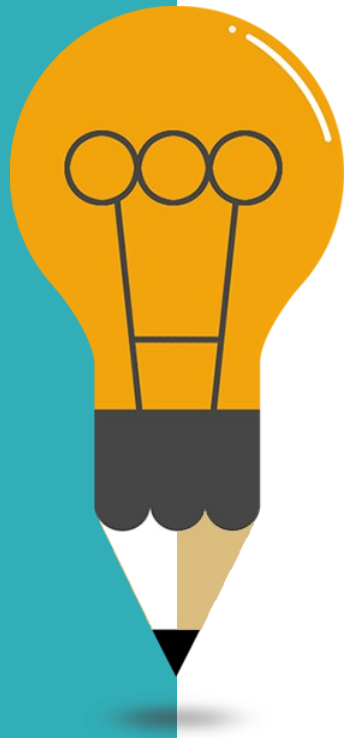
Oracle SQL Basics

Presented By:

Alphalogic Inc.

www.alphalogicinc.com

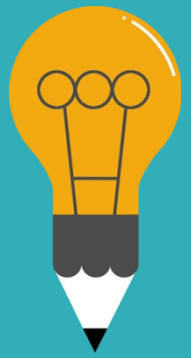
Agenda



01 Software Installation

02 Database Concepts

03 Database Fundamentals



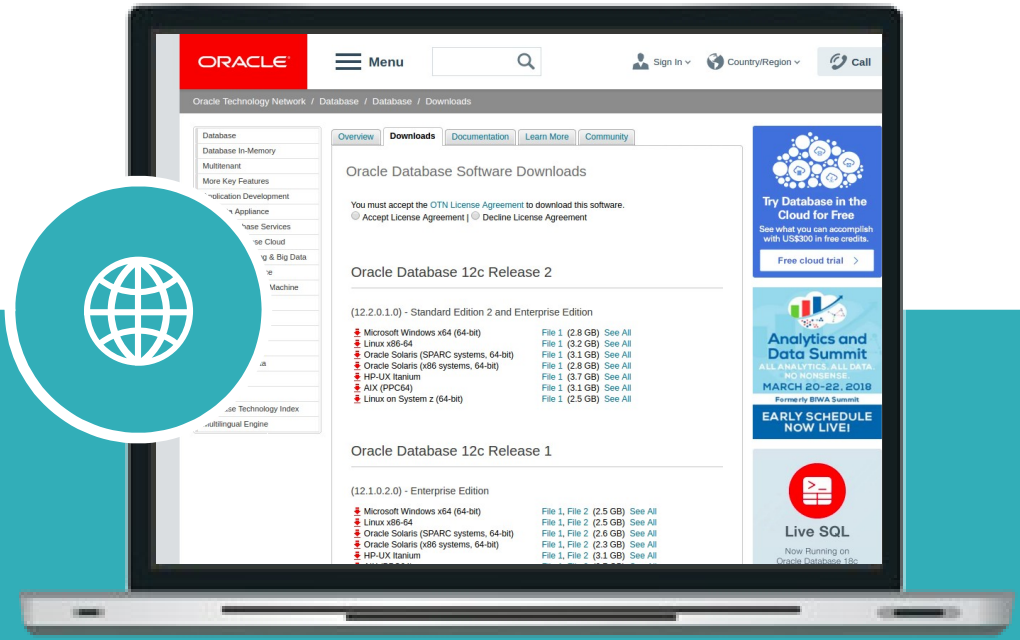
SOFTWARE INSTALLATION

Installation Guide

Install Oracle Database:

Dowanload oracle database from Oracle.com and install it.

Link:
http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.html

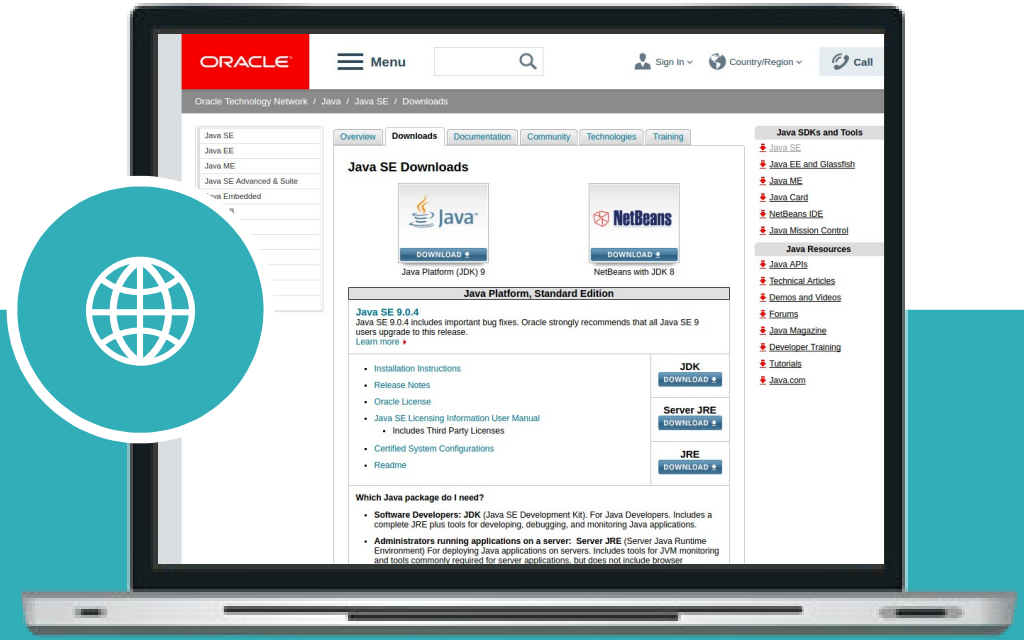


Installation Guide

Install Java SDK:

Download latest Java SDK from Oracle.com and install it.Set Environment Path for SDK.

Link:
http://www.oracle.com/technetwork/java/javase/downloads/index.html

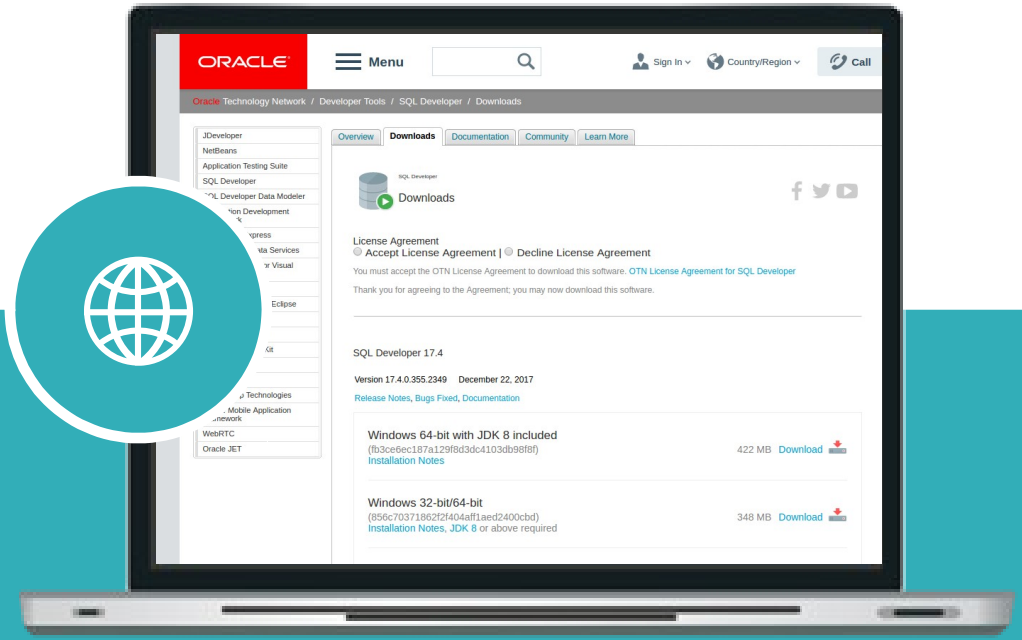


Installation Guide

Install SQL Developer:

Download SQL Developer from Oracle.com and install it. SQL Developer is tool to execute and create SQL Queries.

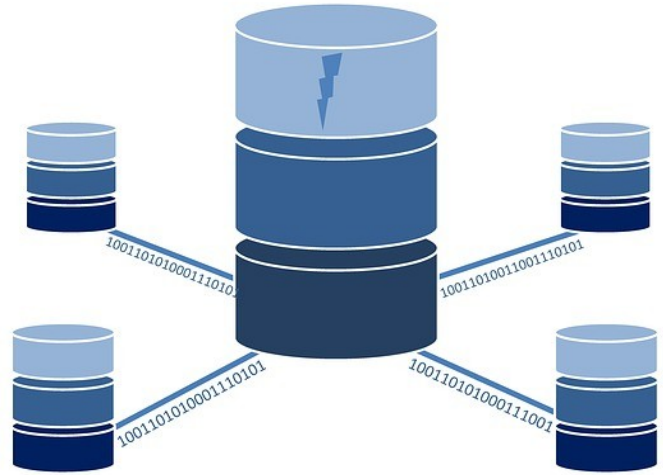
Link:
http://www.oracle.com/technetwork/developer-tools/sql-developer/downloads/index.html



DATABASE CONCEPTS

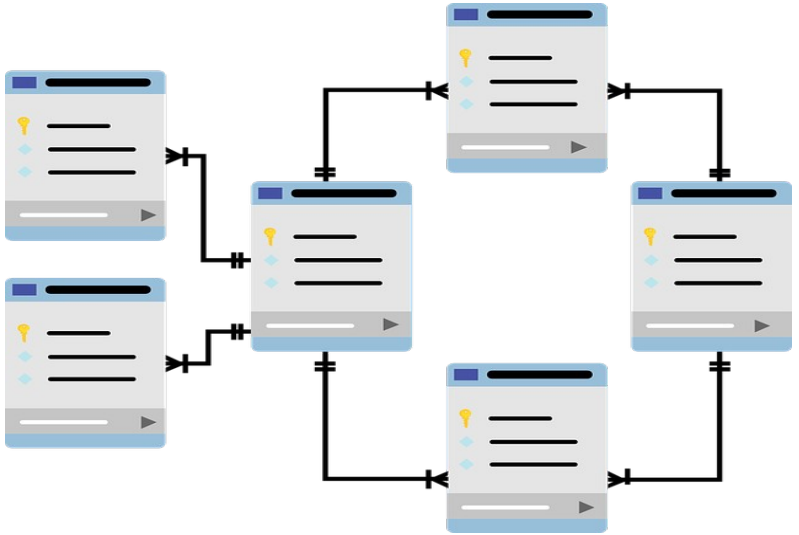
What is Database?

A database is a collection of information that is well oraganised so that it can be easily accessed, managed and updated. It is a repository which stores the tables.



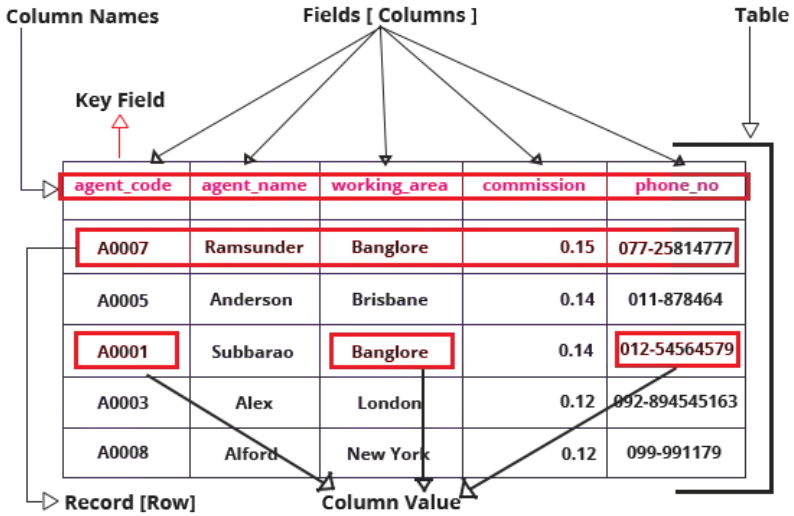
What is Relational Database (RDBMS)?

RDBMS stores the data into collection of tables which might be realted by common fields(columns).



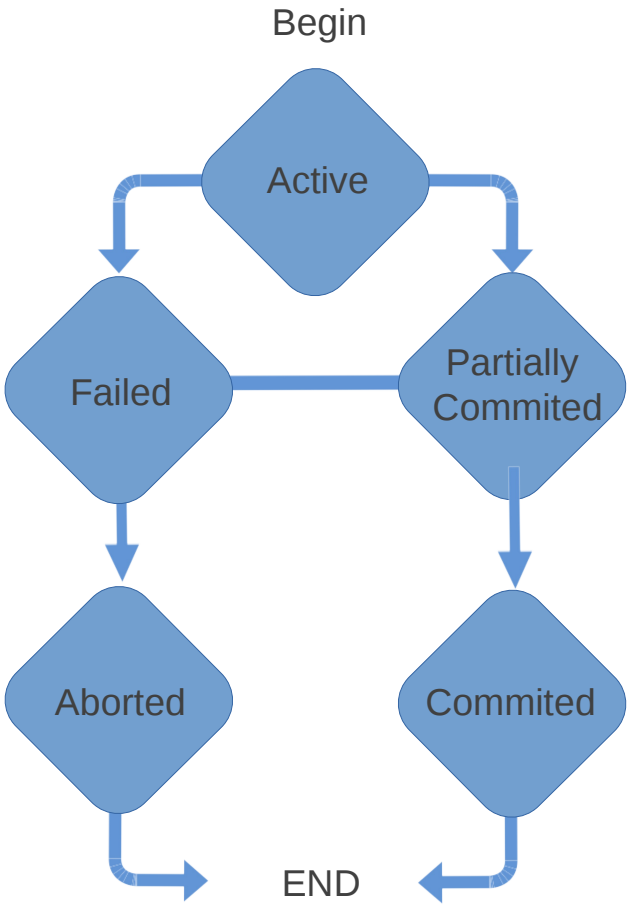
What is a Table?

- A table is a collection of related data held in a structured format within a database
- It consists of Fields(Columns) and Records(Rows)
- Every Column has a datatype- Table follows rules like if a column of number datatype can only hold number values so the data is in structured format



What is Transaction?

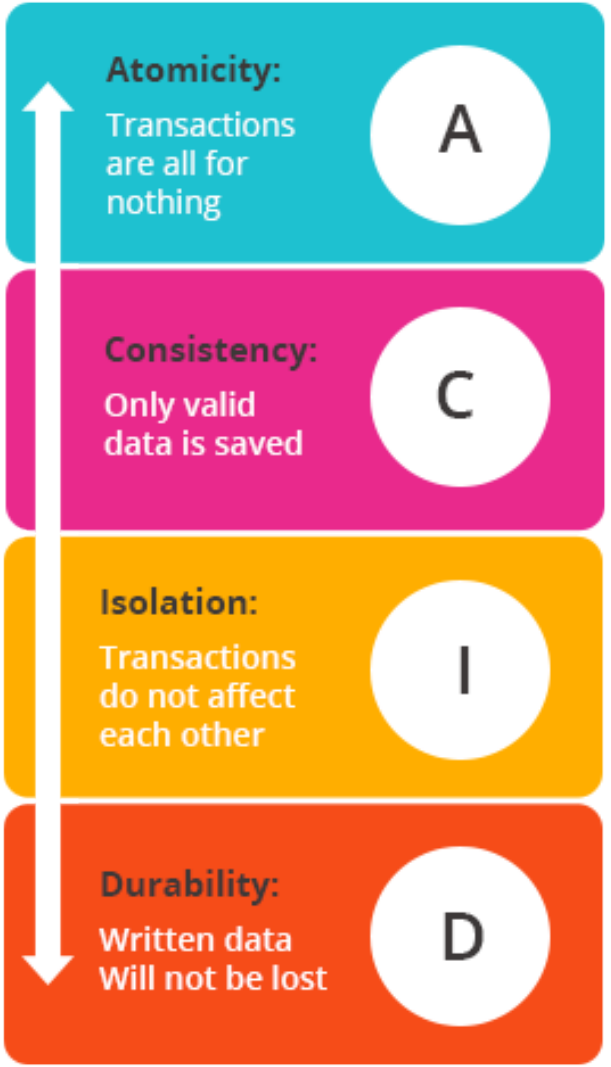
- A transaction comprises of a Unit of work performed within a system against a database and is performed in a reliable way independent of other transactions.
- A transaction is reliable, means if any step during a transaction gets failed then whole transaction would get failed



ACID Properties

ACID refers to the basic properties of a database transaction. All oracle datbase comply with ACID properties.

- **Atomicity**: The 'All' or 'Nothing' property.The entire seq of actions must be completed or aborted.
- **Consistency**: The transaction takes the resource from one steady state to another steady state.
- **Isolation**: A transaction's effect is not visible to other transaction until the transaction is commited. A transaction can not interfere with another transaction
- **Durability**: Chnages made by the commited transaction are permanent and must survive the system failure.



DATABASE FUNDAMENTALS

Enter The Database

How do we interact with a Database?

- SQL- Structured Query Langauge is a computer language used for storing, manipulating data stored in a relational database
- It supports all relational operator
- Can be embedded in any procedural language Used for insert and create data
- Very easy as the English language

Let's See what tables do I own?

- Using CAT data dictionary SELECT * FROM CAT;
- CAT - Catalouge of all the tables owned by the users

Let's Limit The Data

01

Use of Where clause for filtering numeric value

- SELECT * FROM sales WHERE total_amount > 1000;
- SELECT * FROM sales WHERE total_amount != 44;
- SELECT * FROM sales WHERE total_amount^44;
- SELECT * FROM sales WHERE quantity <= 10;

02

Use of Where clause for filtering text value

- SELECT * FROM sales WHERE sales_date = '09-feb-2015';
- SELECT * FROM product WHERE color = 'RED';

03

Use of Where clause for comparing column values

- SELECT * FROM sales WHERE total_amount > sales_amount;

Logical Operators

01

Use of BETWEEN and NOT BETWEEN

- SELECT * FROM sales WHERE total_amount NOT BETWEEN 1 and 100;
- SELECT * FROM sales WHERE total_amount BETWEEN 1 and 100;

02

Use of IN

- SELECT * FROM sales WHERE quantity IN (20,2,10);

03

Use of LIKE

- SELECT * FROM product WHERE product_name LIKE 'Mob%';
- SELECT * FROM product WHERE product_name LIKE '%Mob';
- SELECT * FROM product WHERE product_name LIKE 'Mob_Device';

Logical Operators

04 **Use of ALL**

- SELECT * FROM sales WHERE total_amount > ALL (50,100,200);

05 **Use of ANY**

- SELECT * FROM sales WHERE total_amount > ANY (50,100,200);

06 **Use of NULL**

- SELECT * FROM product WHERE color IS NULL;

07 **Use of AND**

- SELECT * FROM sales WHERE total_amount > 100 AND quantiy < 20;

Arithmetic Operator

1) **Addition (+) :** SELECT 100/ 20 FROM DUAL; --5

2) **Substraction (-) :** SELECT 100+20 FROM DUAL; --120

3) **Multiplication (*) :** SELECT 100-20 FROM DUAL; --80

4) **Division (/) :** SELECT 100*20 FROM DUAL; --2000

5) **Modulus (%) :** SELECT 10%100 FROM DUAL; --10

Let's Sort the Data

Order By Clause

- SELECT sales_date, product_id, order_id, sales_amount, tax_amount FROM sales ORDER BY tax_amount;
- SELECT sales_date, product_id, order_id, sales_amount, tax_amount FROM sales ORDER BY sales_amount, tax_amount;
- SELECT order_id, sales_date, product_id, sales_amount, tax_amount FROM sales ORDER BY order_id DESC;

How NULL values are treated while sorting the Data?

NULL values are treated as very large value by Oracle. So NULL data will sort to the bottom of the sort is in ascending order and to the top of the sort is in descending order.

Set Operators

- Set operators combines the result of two component queries into a single unit. Queries containing the set operators are called compound queries
- The data type of columns should be same to use the set operators
- Types of Set Operators: UNION, UNION ALL, INTERSECT, and MINUS

Set Operators

01

UNION ALL

- SELECT order_id FROM sales UNION ALL SELECT order_id FROM sales_history;

02

UNION

- SELECT order_id FROM sales UNION SELECT order_id FROM sales_history;

03

INTERSECT

- SELECT order_id FROM sales INTERSECT SELECT order_id FROM sales_history;

04

MINUS

- SELECT order_id FROM sales MINUS SELECT order_id FROM sales_history;

Let's group the data

01

Aggregate/Summary Functions

- Aggregate Funcitons returns a single result row based on the group of rows.
- Some of the functions are: MIN(), MAX(), COUNT(), SUM(), AVG()

02

SUM Function

- SELECT sales_date, SUM(total_amount) FROM sales GROUP BY sales_date;

Let's group the data

03

MAX Funciton

- SELECT sales_date, order_id, MAX(total_amount) FROM sales GROUP BY sales_date, order_id;

04

MIN Function with Having Clause

- SELECT sales_date, MIN(total_amount) FROM sales GROUP BY sales_date HAVING MIN(total_amount)< 100;

05

Difference Between Where and Having?

- Where clause is used to filter the detailed result data whereas Having clause is used to filter the aggregated result.

JOINS

Joins are used to join one or more table in database using a common column in tables.

Why JOINS?

To get data from two or more tables in single SQL statement
To avoid the unwanted duplication of data, we split the single table into multiple table and join them on the basis of common columns

Interesting Things

CASE Statements:

The CASE Statements evaluated a single expression and compares it against several potential values, or evaluates multiple boolean expression and choose the first one that is true.

Alias Name:

We can provide different titles to the column name. Spaces are not allowed in an alias name.

If required, then use double quotes : SELECT SUM(AMOUNT) "TOTAL AMOUNT" FROM SALES;

Pseudo Columns in Oracle:

A pseudo column is an Oracle assigned value used in the same context as column value but not stored on disk.

SYSDATE: Returns Current Date

USER: Returns the current timestamp

ROWNUM: It indicates a number indicating the order of the row selected from the table

ROWID: Returns the RowID(Binary Address) of a row in a database table

Data Definition Language DDL

01

Create Table statement

CREATE TABLE movies (Movie_number number, Movie_name varchar2(100), Movie_type varchar2(40), Movie_release_date date);

02

Add column to table

ALTER TABLE movies ADD (movie_language varchar2(30));

03

Modify Column attributes

ALTER TABLE movies MODIFY (movie_type varchar2(50));

04

Drop Table

DROP TABLE movies;

Data Definition Language DDL

05

Insert Values into a table

INSERT INTO movies VALUES (01, 'TERMINATOR', 'ACTION', '12-JAN-2015');
COMMIT;

06

Update a record

UPDATE movies set movie_release_date = '14-jan-2015' WHERE movie_number = 101;
COMMIT;

07

Delete a record

DELETE from movies WHERE movie_name = 'RUSH HOUR';
COMMIT;

08

Truncate Statement

TRUNCATE TABLE SALES;

Data Definition Language DDL

Difference between DELETE and TRUNCATE?

01 ROLLBACK after DELETE can work, but not after TRUNCATE.

02 TRUNCATE auto commits.

03 DELETE generates a small amount of REDO space and a large amount of UNDO space but TRUNCATE generates neither of these two.

Let's Put Some Restrictions

Why constraints?

Constraints apply the specific rule to data, ensuring the data confirms the requirement defined.

Example: NOT NULL, UNIQUE, Primary Key, Check, Foreign Key

CHECK:

Check constraint validates that value in a given column meets specific criteria.

```
CREATE TABLE movies (Movie_number number, Movie_name varchar2(100),
Movie_type varchar2(40)  CHECK  (movie_type IN ('ACTION', 'COMEDY')),
Movie_release_date date);
```

Let's Put Some Restrictions

Foreign Key:

A foreign Key constraint is used to enforce a relationship between two tables

CREATE TABLE movies (Movie_number number, Movie_name varchar2(100), Movie_type varchar2(40), Movie_release_date date, Movie_director_number number REFERENCES director(director_number));

VIEWS

What is a VIEW?

A view is simply the representation of a SQL statement that is stored in memory so that it can be easily reused.

- A view gives a look of a table as defined by the select statement in the view definition
- A view does not store data separately
- Only the definition(SQL statement) of the view is stored
- The data is retrieved from the underlying table based on the view definition
- Advantages:
  - Security - restrict the access of complete data to all
  - Abstraction - Hiding complex logic and just displaying the required output

VIEWS

Create a View

```
CREATE VIEW SALES_MOBILE AS SELECT S.SALES_DATE,  S.ORDER_ID,
S.QUANTITY, S.UNIT_PRICE,  S.TOTAL_AMOUNT,  P.PRODUCT_NAME,
P.PRODUCT_CATEGORY FROM SALES S, PRODUCT P  WHERE
S.PRODUCT_ID = P.PRODUCT_ID AND PRODUCT_CATEGORY = 'Mobile';
```

Modify View

```
CREATE OR REPLACE VIEW SALES_MOBILE AS SELECT S.SALES_DATE,
S.ORDER_ID, S.QUANTITY,  S.UNIT_PRICE,  S.TOTAL_AMOUNT,
P.PRODUCT_NAME, P.PRODUCT_CATEGORY,     S.PRODUCT_ID FROM
SALES S, PRODUCT P  WHERE S.PRODUCT_ID = P.PRODUCT_ID  AND
PRODUCT_CATEGORY = 'Mobile';
```

Drop a View

```
DROP view SALES_MOBILE;
```

Other Database Objects

Synonyms

- Synonym is an alternative name for a table, view, sequence, procedure, stored function
- Syntax: CREATE SYNONYM inventory_data FROM SALES;

Sequences

- A Sequence is an object in Oracle that is used to generate a number series(sequence).
- This can be useful when you need to create a Unique number to act as a primary key.
- Syntax: CREATE SEQUENCE VA_RECORD_ID MIN VALUE 1, MAX VALUE 999999, START_WITH , INCEREMENT BY 1, CACHE 10;
- NEXTVAL is used to get next value of sequence and CURRVAL is used to get the current value.

Giving Permissions To Other Users

GRANT

- GRANT command can be used to grant schema object privileges to the role or to the user
- WITH GRANT OPTION enables the user to pass on the privilege to other user or role.
- Syntax: GRANT SELECT ON SALES TO SCOTT WITH GRANT OPTION;

REVOKE

- REVOKE command can be used to take back the granted privileges to the user or role
- Syntax: REVOKE ALL ON SALES FROM SCOTT;

Sub Queries

Sub Suery

- A sub query is a query within a query which can return one or more rows. A subquery executes inner query before the main query

Multiple Column Subqueries

- Pair wise Comparision: Values compared in pair
  - **Syntax**: SELECT sales_date, order_id, customer_id FROM SALES WHERE (product_id, unit_price) IN (SELECT product_id, unit_price FROM SALESPERSON where sales_date='01-Jan-2009');
- Non Pairwise Comparision: Values are compared individually

Sub Queries

WITH Clause

To hold the result of a SQL statement using a WITH clause and use it in multiple SQL Statement.

- Syntax: WITH st as (SELECT * FROM SALES_TOTAL) SELECT * FROM SALES s, st WHERE s.sales_date=st.sales_date;

Scaler Subquery

Scaler sub queries will allow treating the output of a sub query as a column or even an expression within a select statement.

- It must return only one row and one column.
- Syntax: SELECT s.sales_date, s.id, (SELECT SUM(total_sakes) FROM SALES) as sales_total from SALES s;

Sub Queries

Corelated Subquery

Correlated sub query is a subquery that uses the values from the outer query and is evaluated once for each row processed by the outer query.

- Syntax: SELECT * FROM SALES x where total_amt>(SELECT AVG(sales_amt) FROM SALES y WHERE y.c_id=x.c_id);

Index

Index

An index is a performance tuning method of allowing faster retrieval of records

Properties:

- Indexes enable faster data acess
- Index stores column values and their location
- The index can be created on multiple columns

DROP Index:

DROP INDEX <index_name>;

Index

Unique Index

A unique Index is an Index no duplicate values are allowed

- Unique Index will not accept duplicate values
- It can have null values until and unless restricted
  - Syntax: CREATE UNIQUE INDEX cust_idx ON CUSTOMER(cust_id);

Rename Index

ALTER INDEX <index_old_name> RENAME TO new_name_of_index>;

Thank you