



## Internal Assessment Test III - Aug 2024

Sub	):	Database Ma	nagement S	System			Sub Code:	BCS/BAD 403	Branch	n: AIND	S/CS	(DS)
Dat	e:	07/08/2024	Duration:	90 minutes	Max Marks:	50	Sem		IV	-	0	BE
				Answer any	FIVE Question	15	-			MARK S	СО	RBT
1	a Describe the database inconsistency problems: Lost update, dirty read and blind write?									5	CO1	L1
1	b	With a neat dia		5	CO1	L2						
	а	Check whether the below schedule is conflict or not. { $b2$ , $r2(X)$ , $b1$ , $r1(X)$ , $wl(X)$ , $rl(Y)$ , $wl(Y)$ , $w2(X)$ , $e1$ , $c1$ , $e2$ , $c2$ ).								5	CO1	L1
2	b	What is 2PL? Explain with an example.								5	CO3	L2
3		How do you detect a deadlock during concurrent transaction execution? Explain Deadlocks an Starvation in 2PL.								10	CO3	L2
4		Explain the various database recovery techniques, with examples.								10	CO3	L3
5		Explain Binary locks and shared locks with algorithms.								10	CO1	L2
6		What is the CA tolerance) are 1			e three propertie systems?	s (co	onsistency, ava	ailability, parti	ition	10	CO6	L3

CI



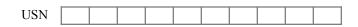
HOD

USN					

	1 D.C. +	
-630 YE	ARS *	
A.	$\mathcal{A}$	
COLERRY COLERRY		
- ES 🤍 🚬		
*.		MRIT
CMB INS	LITUTE OF TECHNO	LOGY BENGALLIR

Internal	Assessment	Test III	– Aug 2024
meenan	1 1000000000000000000000000000000000000	1000 111	1146 2021

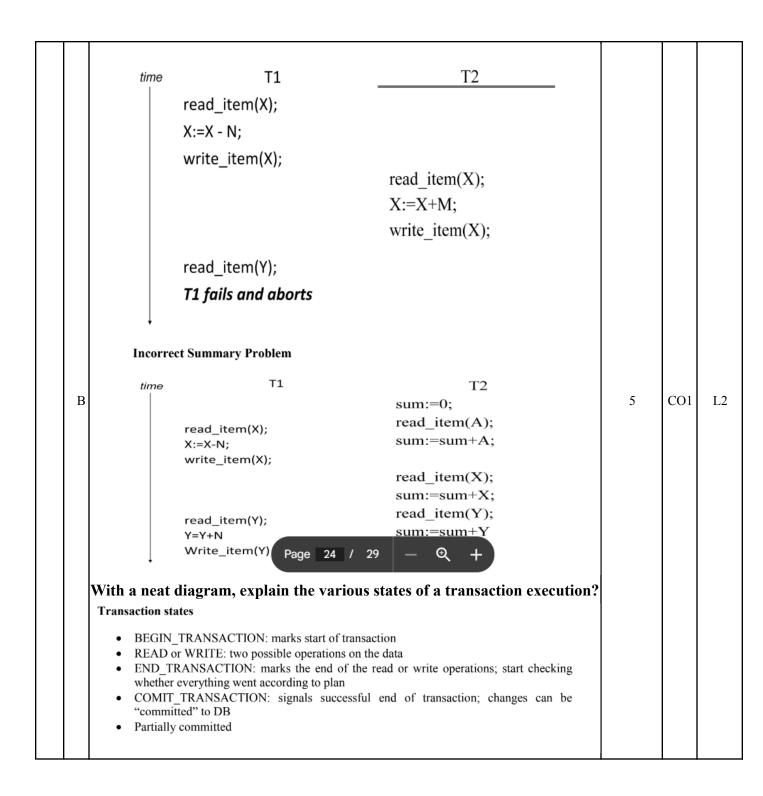
Sub	:	Database Ma	anagement S	System			Sub Code:	BCS/BAD 403	Branch	AIND	S/CS	(DS)
Dat	e:	07/08/2024	Duration:	90 minutes	Max Marks:	50	Sem		IV		0	BE
				Answer any	FIVE Question	<u>IS</u>				MARK S	СО	RBT
	a Describe the database inconsistency problems: Lost update, dirty read and blind write?										CO1	L1
1	b	With a neat diagram, explain the various states of a transaction execution?									CO1	L2
	a	Check whether the below schedule is conflict or not. { $b2$ , $r2(X)$ , $b1$ , $r1(X)$ , $wl(X)$ , $rl(Y)$ , $wl(Y)$ , $w2(X)$ , $e1$ , $e2$ , $e2$ ).								5	CO1	L1
2	b	What is 2PL?	Explain with	n an example.						5	CO3	L2
3		How do you detect a deadlock during concurrent transaction execution? Explain Deadlocks and Starvation in 2PL.							cks and	10	CO3	L2
4		Explain the var	rious databa	se recovery te	chniques, with e	xam	ples.			10	CO3	L3
5		Explain Binary locks and shared locks with algorithms.								10	CO1	L2
6		What is the CA tolerance) are n			e three propertie <i>systems</i> ?	s (cc	onsistency, ava	ailability, parti	ition	10	CO6	L3

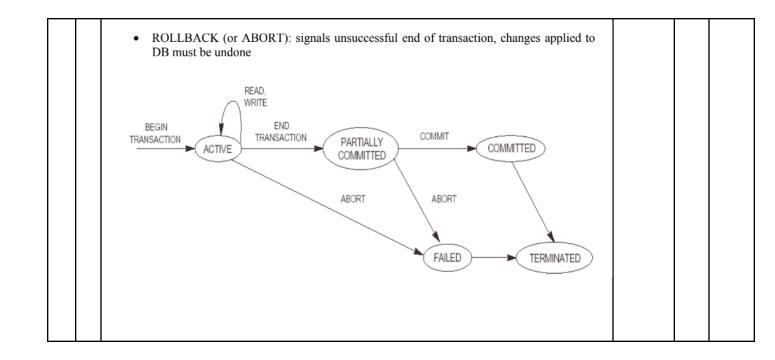


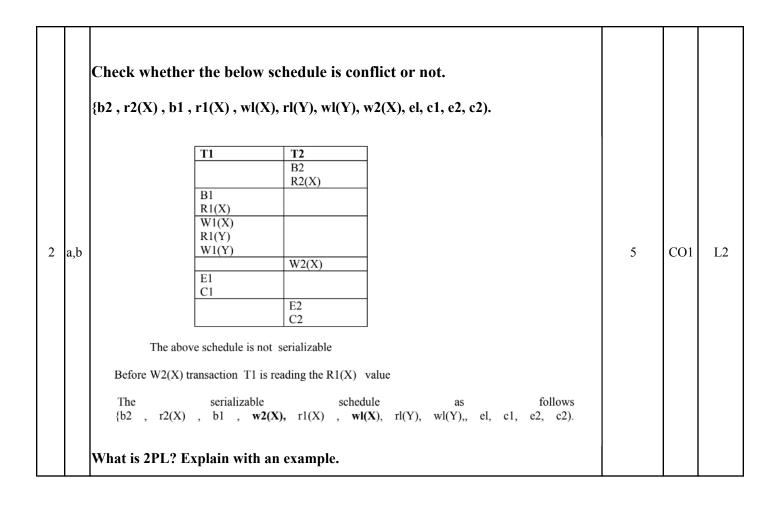


## Internal Assessment Test III - Aug 2024

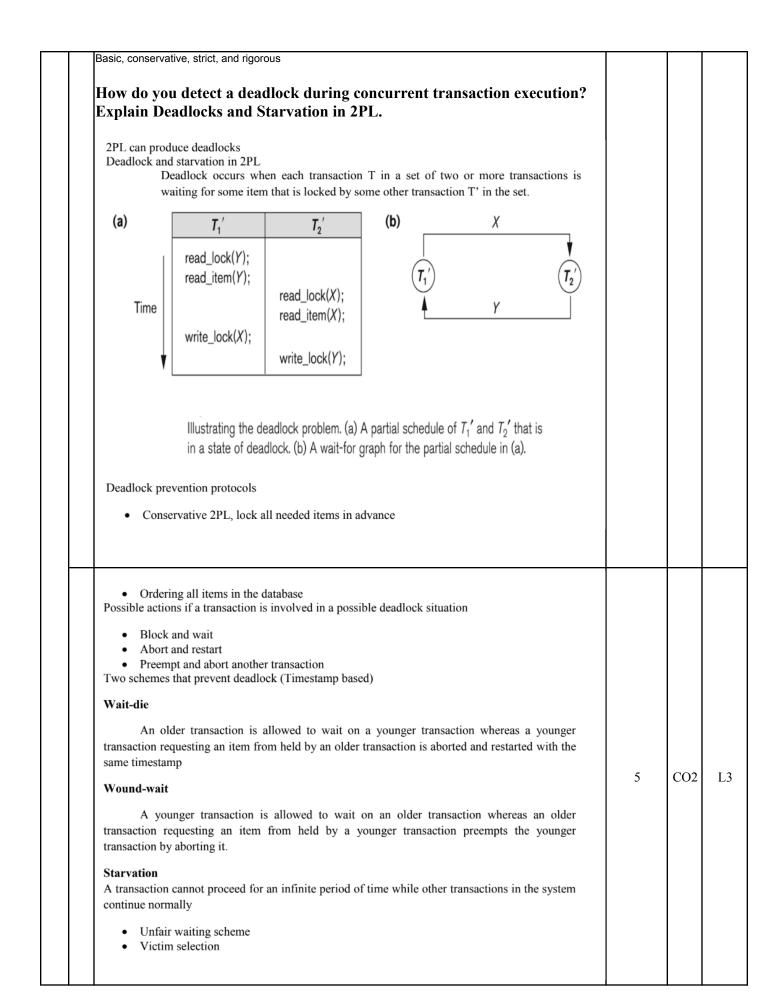
Sub:	Database M	anagement	System			Sub Code:	BCS/BAD 403	Branch:	AIND	S/CS	(DS)
Date:	11/07/2024	Duration:	90 minutes	Max Marks:	50	Sem		IV		0	BE
			Answer any	FIVE Question	<u>IS</u>				MARK S	со	RBT
1	and blind y DBMS has a despite concu Three proble 1. 2. 3. Lost Update	write? Concurrency urrent executio ms are occurs of The lost upd The tempora Incorrect sur Problem ime read_i X:=X - write_ read_i Y:=Y +	<u>Control</u> sub-sy n of transactions during concurrer ate problem ry update (dirty nmary problem T1 (tem(X); N; _item(X); tem(Y); N; _item(Y);	read) problem read X:=3	_iter X+N	e remains in co T2 n(X);		<sup>7</sup> read	5	CO1	L1







Two-Phase Locking (2PL) Protocol		
Transaction is said to follow the <i>two-phase-locking protocol</i> if all locking operations precede the <i>first</i> unlock operation		
Expanding (growing) phase		
Shrinking phase		
During the shrinking phase no new locks can be acquired		
Downgrading ok		
Upgrading is not		
2PL Example		
$\begin{array}{cccc} T1' & T2' \\ read_lock(Y); & read_lock(X); \\ read_item(Y); & read_item(X); \\ write_lock(X); & write_lock(Y); \\ unlock(Y); & unlock(X); \\ read_item(X); & read_item(Y); \\ X:=X+Y; & Y:=X+Y; \\ write_item(X); & write_item(Y); \\ unlock(X); & Page 26 / 29 & unlock(Y); \\ \end{array}$		
Both T1' and T2' follow the 2PL protocol Any schedule including T1' and T2' is guaranteed to be serializable Limits the amount of concurrency Two-phase locking protocol (2PL) All lock operations precede the first unlock operation Expanding phase and shrinking phase Upgrading of locks must be done in expanding phase and downgrading of locks must be done in shrinking phase If every transaction in a schedule follows 2PL protocol then the schedules is guaranteed to be serializable. Variants of 2PL		



			-	
	c)Explain the various database recovery techniques, with examples.			
	· · · · · · · · · · · · · · · · · · ·			
	<ul> <li>Keeps information about operations made by transactions: <ul> <li>Before-image (undo entry) of updated items</li> <li>After-image (redo entry) of updated items</li> </ul> </li> <li>Enables restoring a consistent state after non-catastrophic failure (forward/backward).</li> <li>Alternatives: <ul> <li>undo/no-redo</li> <li>no-undo/redo</li> <li>no-undo/no-redo.</li> </ul> </li> <li>Write-Ahead Logging (WAL) </li> <li>(1) No overwrite of disk data before undo-type log records are forced to disk.</li> </ul>			
	<ul> <li>(2) Both undo- and redo-type log records</li> <li>(= before- and after-images) must</li> <li>be forced to disk before end of commit.</li> <li>Backup</li> </ul>			
3	Copy of database on archival storage (off-line, often on tape). Enables partial recovery from <i>catastrophic</i> failures: For <i>committed</i> transactions: Load backup and apply redo operations from the log (if the log survived). <i>Non-committed</i> transactions must be restarted (= re-executed). <b>Cache</b>	10	CO2	L4
	<ul> <li>In-memory buffer for database pages.</li> <li>A <i>directory</i> (page table) keeps track of pages in cache.</li> <li>Page-replacement strategy needed, e.g.</li> <li><i>FIFO</i> (First-In-First-Out), or</li> <li><i>LRU</i> (Least Recently Used)</li> <li><i>Dirty bit</i> tells for each page, if it has changed</li> <li><i>Flushing</i> means (force-)writing buffer pages to disk.</li> <li><i>Pin/unpin bit</i> tells if the page can be written</li> <li><b>Rollback</b></li> <li>At failure, apply undo-type log records (before-images) to updated items.</li> <li>A recoverable schedule may allow <i>cascading rollback</i>.</li> <li>Most practical protocols avoid cascading rollbacks. Then no <i>read-entries</i> are required in the log (no dirty reads).</li> </ul>			
	(no dirty reads). Explain binary locks and shared locks with algorithms?			

```
Binary locks and shared locks are two important concepts in concurrent programming
and database systems. Let's explore each of them:
    1. Binary Locks (Exclusive Locks):
Binary locks, also known as exclusive locks or mutex locks, allow only one process or
thread to access a shared resource at a time. When a process acquires a binary lock, it
has exclusive access to the resource, and all other processes must wait until the lock is
released.
Algorithm for Binary Lock:
struct BinaryLock {
  bool locked = false;
function acquire(BinaryLock lock) {
  while (true) {
     if (!lock.locked) {
       lock.locked = true;
       return;
     }
     // Wait or yield to other processes
  }
function release(BinaryLock lock) {
  lock.locked = false;
    2. Shared Locks (Read-Write Locks):
Shared locks allow multiple processes to read a shared resource simultaneously, but
ensure exclusive access for writing. This is useful when read operations are more
frequent than write operations.
There are two types of locks in this system:
        Read lock: Multiple processes can hold read locks simultaneously.
        Write lock: Only one process can hold a write lock, and no read locks can be
    •
        held simultaneously.
Algorithm for Shared Lock:
struct SharedLock {
  int readers = 0;
  bool writer = false;
```

```
BinaryLock readLock;
  BinaryLock writeLock;
function acquireReadLock(SharedLock lock) {
  acquire(lock.readLock);
  if (lock.readers == 0) {
    acquire(lock.writeLock);
  }
  lock.readers++;
  release(lock.readLock);
function releaseReadLock(SharedLock lock) {
  acquire(lock.readLock);
  lock.readers--;
  if (lock.readers == 0) {
    release(lock.writeLock);
  }
  release(lock.readLock);
function acquireWriteLock(SharedLock lock) {
  acquire(lock.writeLock);
function releaseWriteLock(SharedLock lock) {
  release(lock.writeLock);
What is the CAP Theorem? Which of the three properties (consistency, availability,
partition tolerance) are most important in NOSQL systems?
The CAP Theorem, also known as Brewer's Theorem, is a fundamental concept in
distributed computing systems, particularly relevant to distributed databases and
NoSQL systems. It states that in a distributed data store, it is impossible to
simultaneously guarantee all three of the following properties:
1. Consistency (C): All nodes see the same data at the same time. In other words, a
read operation will return the most recent write operation's value, no matter which
node it contacts.
```

contains the	ty (A): Every request receives a response, without guarantee that it most recent version of the data. The system remains operational and can
respond to re	quests even if some nodes are down.
	Colerance (P): The system continues to operate despite network partitions between nodes.
	asserts that a distributed system can only guarantee two out of these three any given time.
environments partitions are Therefore, N	systems, which are often designed for large-scale, distributed s, Partition Tolerance is generally considered non-negotiable. Network e a reality in distributed systems, and the ability to handle them is crucial. NoSQL databases typically have to choose between Consistency and when network partitions occur.
Most NoSQI Consistency.	L systems prioritize Availability and Partition Tolerance (AP) over strict Here's why:
	oSQL databases are often used in scenarios requiring high scalability. availability allows the system to continue functioning even when some reachable.
synchronize a	nce: Strict consistency can introduce latency, as the system needs to all nodes before responding to requests. Many NoSQL use cases prioritize responses over perfect consistency.
	Requirements: Many applications using NoSQL can tolerate eventual where the system will become consistent over time, rather than requiring onsistency.
more comm	ic Distribution: For globally distributed databases, network partitions are on, and maintaining strict consistency across all regions can be and performance-intensive.

However, it's important to note that this is a generalization. Different NoSQL databases make different trade-offs:

- Cassandra and DynamoDB, for instance, typically favor Availability and Partition Tolerance (AP).

- MongoDB and HBase lean more towards Consistency and Partition Tolerance (CP) in their default configurations.

- Some modern distributed databases like Google's Spanner aim to provide all three properties most of the time, only sacrificing availability during rare network partitions.

The choice between consistency and availability often depends on the specific use case. For example, a financial system might prioritize consistency over availability, while a content delivery system might prioritize availability over strict consistency.

In summary, while Partition Tolerance is crucial for NoSQL systems, the choice between Consistency and Availability is often application-specific. However, many NoSQL systems are designed with a bias towards Availability and Partition Tolerance, with various mechanisms to provide different levels of eventual consistency.

5	Module	Dept	Lecturer	Text					
5	M1	D1	Ll	T1					
(12)	M1	D1	Ll	T2					
6.	🕅 M2	D1	Ll	T1					
	M2	D1	Ll	T3					
Ø	M3	D1	L2	T4					
1	M4	D2	L3	T1					
	M4	D2	L3	T5					
	M5	D2	L4	T6					
•				Third Norn	nal Form (3NF), w	e'll follow the norr	malization process:		
-	-		dependencies.	E (1N					
			n First Normal l Normal Form		F).				
			Normal Form						
	dentify Funct			(01(1))					
•	From the gi	-							
٠	Module det	ermines <b>E</b>	ept and Lectur	er.					
٠				termine all	attributes uniquely	, as each combinat	tion of <b>Module</b> and		
	Text unique	2							
•			e following fun	ctional depe	endencies:				
	$\rightarrow$ Dept, Lect		T (						
	Text $\rightarrow$ Dept								
• tep 3: 5	Second Norma			ll non-key a	ttributes are fully	functionally denon	dent on the primary	7	
•				-	ependencies on a c		sent on the prinary	, ,	
•	•		Key: (Module, T	-		omposite key.			
•					ove partial depende	encies			
•	Decomposi		-		· · · · · · · · · · · · · · · · · · ·				
able 1:	Modules			able 2: Mo	dule_Text			10	CC
		-	_					10	100
35.	1.   D ( )								
Mod	-	Lecture	r 🗌	Modu	le Text	7			
M	1 D1	L1		$\mathbf{M1}$	<b>T1</b>	-			
	1 D1			M1 M1	T1 T2	_			
M M	1 D1 2 D1	L1 L1		$\mathbf{M1}$	<b>T1</b>	_			
M M M	1 D1 2 D1 3 D1	L1 L1 L2	<u>r</u>	M1 M2 M2 M3	T1 T2 T1 T3 T4				
M M M M	$ \begin{array}{c ccccc} 1 & D1 \\ 2 & D1 \\ 3 & D1 \\ 4 & D2 \\ \end{array} $	L1 L1 L2 L3		M1 M2 M2 M3 M4	T1 T2 T1 T3 T4 T1				
M M M	$ \begin{array}{c cccc} 1 & D1 \\ 2 & D1 \\ 3 & D1 \\ 4 & D2 \\ \end{array} $	L1 L1 L2		M1 M2 M2 M3	T1 T2 T1 T3 T4				
M M M M	1 D1 2 D1 3 D1 4 D2 5 D2 Third Norma	L1 L1 L2 L3 L4	3NF)	M1 M1 M2 M3 M4 M4 M5	$\begin{array}{c} {\rm T1} \\ {\rm T2} \\ {\rm T1} \\ {\rm T3} \\ {\rm T4} \\ {\rm T1} \\ {\rm T5} \\ {\rm T6} \end{array}$		t only on the prime	arv	
M M M M	1         D1           2         D1           3         D1           4         D2           5         D2           Third Normal A table is in	L1 L1 L2 L3 L4 al Form ( n 3NF if i	3NF) t is in 2NF and	M1 M1 M2 M2 M3 M4 M4 M5	T1 T2 T1 T3 T4 T1 T5 T6	onally dependent	t only on the prima	ary	
M M M M	1         D1           2         D1           3         D1           4         D2           5         D2           Third Normal A table is in key and the	L1 L1 L2 L3 L4 al Form ( n 3NF if i ere are n	3NF) t is in 2NF and o transitive dep	M1 M1 M2 M2 M3 M4 M4 M5	T1 T2 T1 T3 T4 T1 T5 T6		t only on the prima		
M M M M Step 4:	1         D1           2         D1           3         D1           4         D2           5         D2           Third Normal A table is in key and the	L1 L1 L2 L3 L4 al Form ( n 3NF if i ere are n odules is	3NF) t is in 2NF and o transitive dep s already in 3N	M1 M1 M2 M2 M3 M4 M4 M5	T1 T2 T1 T3 T4 T1 T5 T6				
M M M M Sitep 4:	1     D1       2     D1       3     D1       4     D2       5     D2   Third Normal A table is in key and the Table 1: M the primary	L1 L1 L2 L3 L4 al Form ( n 3NF if i ere are n odules is / key (Mc	3NF) t is in 2NF and o transitive dep s already in 3N idule).	M1 M1 M2 M2 M3 M4 M4 M5	T1 T2 T1 T3 T4 T1 T5 T6	es (Dept, Lecture			
M M M M Step 4:	1     D1       2     D1       3     D1       4     D2       5     D2   Third Norma A table is in key and the Table 1: M the primary Table 2: M primalized Table 1: M	L1 L1 L2 L3 L4 al Form ( a 3NF if i ere are n odules is / key (Mc odule_T bles:	3NF) t is in 2NF and o transitive dep s already in 3N idule).	M11 M12 M22 M33 M44 M4 M5 all the attr pendencie: IF since all ince there	T1 T2 T1 T3 T4 T1 T5 T6	es (Dept, Lecture dependencies.			
M M M M Step 4:	1     D1       2     D1       3     D1       4     D2       5     D2   Third Norma A table is in key and the Table 1: M the primary Table 2: M	L1 L1 L2 L3 L4 al Form ( a 3NF if i ere are n odules is / key (Mc odule_T bles:	3NF) t is in 2NF and o transitive dep s already in 3N idule).	M11 M12 M22 M33 M44 M4 M5 all the attr pendencie: IF since all ince there	T1 T2 T1 T3 T4 T1 T5 T6	es (Dept, Lecture dependencies.			
M M M Step 4: •	1     D1       2     D1       3     D1       4     D2       5     D2   Third Norma A table is in key and the Table 1: M the primary Table 2: M primalized Table 1: M	L1 L2 L3 L4 al Form ( n 3NF if i ere are n odules is / key (Mc odule_T bles: les	3NF) t is in 2NF and to transitive dep s already in 3N dule). ext is in 3NF s	M11 M12 M22 M3 M4 M4 M5 all the attr pendencies IF since all ince there Tabl	T1 T2 T1 T3 T4 T1 T5 T6	es (Dept, Lecture dependencies.			
M M M M M Step 4:	1     D1       2     D1       3     D1       4     D2       5     D2   Third Normal A table is in key and the Table 1: M the primary Table 2: M formalized Table 1: Modu	L1 L2 L3 L4 al Form ( n 3NF if i ere are n odules is / key (Mc odule_T bles: les	3NF) t is in 2NF and to transitive dep s already in 3N dule). ext is in 3NF s	M11 M12 M22 M33 M44 M45 all the attr bendencies if since all ince there Tabl	$\begin{array}{ c c c } T1 \\ T2 \\ T1 \\ T3 \\ T4 \\ T1 \\ T5 \\ T6 \\ \hline \end{array}$	es (Dept, Lecture dependencies.			
M M M M M M tep 4: • • • • • • • • • • • • • • • • • • •	1     D1       2     D1       3     D1       4     D2       5     D2   Third Norma A table is in key and the Table 1: Mo the primary Table 2: M ormalized Ta ble 1: Modu dule Dept	L1 L2 L3 L4 al Form ( n 3NF if i ere are n odules is / key (Mc odule_T bles: les Lectur L1	3NF) t is in 2NF and to transitive dep s already in 3N dule). ext is in 3NF s	M1 M1 M2 M2 M3 M4 M4 M5 all the attu oendencie: IF since all ince there Tabl	$\begin{array}{ c c c } & T1 \\ T2 \\ T1 \\ T3 \\ T4 \\ T1 \\ T5 \\ T6 \\ \hline \end{array}$ ributes are functions.	es (Dept, Lecture dependencies.			
M M M M M M tep 4: • • • • • • • • • • • • • • • • • • •	1     D1       2     D1       3     D1       4     D2       5     D2   Third Normal A table is in key and the table is in key and the Table 1: Module Table 2: Mormalized Table 1: Module Table 1: Module Dept 11       ble 1: Module     Dept 11	L1 L2 L3 L4 al Form ( n 3NF if i ere are n odules is ( key (Mo odule_T bles: les Lectu: L1 L1	3NF) t is in 2NF and to transitive dep s already in 3N dule). ext is in 3NF s	M1 M1 M2 M2 M3 M4 M4 M5 If since all ince there Tabl	$\begin{array}{c c} T1 \\ T2 \\ T1 \\ T3 \\ T4 \\ T1 \\ T5 \\ T6 \\ \hline \\ ributes are functions \\ s. \\ I non-key attribute \\ are no transitive \\ are no transitive \\ \hline \\ $	es (Dept, Lecture dependencies.			
M M M M M M tep 4: • • • • • • • • • • • • • • • • • • •	1     D1       2     D1       3     D1       4     D2       5     D2   Third Normal A table is in key and the Table 1: M the primary Table 2: M formalized Table 1: Module ormalized Table 1: Module       dule     Dept 11       10     D1       12     D1       13     D1	L1 L2 L3 L4 al Form ( n 3NF if i ere are n odules is / key (Mc odule_Tr bles: les Lectu: L1 L1 L2	3NF) t is in 2NF and to transitive dep s already in 3N dule). ext is in 3NF s	M11 M12 M22 M33 M44 M4 M4 M5 all the attr bendencies ince there Tabl	$\begin{array}{ c c c } & T1 \\ T2 \\ T1 \\ T3 \\ T4 \\ T1 \\ T5 \\ T6 \\ \hline \end{array}$	es (Dept, Lecture dependencies.			
M M M M M Step 4: • • • • • • • • • • • • • • • • • • •	1     D1       2     D1       3     D1       4     D2       5     D2   Third Normal A table is in key and the table is in key and the Table 1: Module Table 2: Mormalized Table 1: Module Table 1: Module Dept 11       ble 1: Module     Dept 11       dule     Dept 11       11     D1       12     D1	L1 L2 L3 L4 al Form ( n 3NF if i ere are n odules is ( key (Mo odule_T bles: les Lectu: L1 L1	3NF) t is in 2NF and to transitive dep s already in 3N dule). ext is in 3NF s	M1 M1 M2 M2 M3 M4 M4 M5 If since all ince there Tabl	$\begin{array}{c c} T1 \\ T2 \\ T1 \\ T3 \\ T4 \\ T1 \\ T5 \\ T6 \\ \hline \\ ributes are functions \\ s. \\ I non-key attribute \\ are no transitive \\ are no transitive \\ \hline \\ $	es (Dept, Lecture dependencies.			

1 1 1		

ii) Primary key :- A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table. iii) Primary keys must contain UNIQUE values, and cannot contain NULL values. A table can have only ONE primary keys; and in the table, this primary key can consist of single or multiple columns. SQL PRIMARY KEY on CREATE TABLE • The following SQL creates a PRIMARY KEY on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, FirstName varchar(255), Age int, PRIMARY KEY on ALTER TABLE To create a PRIMARY KEY constraint on the 'ID' column when the table is already created, use the following SQL: ALTER TABLE Persons DROP a PRIMARY KEY constraint on the 'ID' column when the table is already created, use the following SQL: ALTER TABLE Persons DROP a PRIMARY KEY constraint To drop a PRIMARY KEY constraint, use the following SQL: ALTER TABLE Persons DROP PRIMARY KEY: Or create a PRIMARY KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The following SQL creates a FOREION KEY on the "PersonID" column when the "Orders" table is created: CREATE TABLE Orders Order D orders a FOREION KEY constraint on the "PersonID" column when the "Orders" table is already created, use the following SQL: CREATE TABLE Orders CREATE TABLE Orders ALTER TABLE Orders ALTER TABLE Orders ADD FOREION KEY constraint on the "PersonID" column when the "Orders" table is already created, use	) NOT NULL :- Ensures that a co	umn cannot have a NULL value				
The PRIMARY KEY constant unquely identifies each record in a table.     Primary key must contain NULL values.     Atable can here any OAE party hey, and in the table. The primary key can contain the use the second contain NULL values.     Atable can here any OAE party hey, and in the table. The primary key can contain the use the second contain NULL values.     Atable can here any OAE party hey and in the table. The primary key can contain the use the "Desons" table is created:     CERET FABLE Persons (         ID ref NULL,         Labelane worthing F3D. Contain PERSON FX KEY (DI);     Sel PRIMARY KEY CON;     PRIMARY KEY (DI);     PRIMARY KEY CON;     PRIMARY KEY CON;     PRIMARY KEY CON;     PRIMARY KEY CON;     PRIMARY KEY Constraint, use the following SOL:     Attract PRIMARY KEY Constra	,					
A hole on hase only CAP primary keys and its black, Bap framely are can cardial of angle or multiple colume.     Aller MARKY KEY on CREAT FAILE     Development of the Control of the Column when the "Persons" table is created:         OFAPT FAILE     Development of the Control of the Column when the "Persons" table is created:         OFAPT FAILE     Development of the Column of the Column when the "Development table is created:         OFAPT FAILE Persons         OFAPT FAILE     Development of the Column when the table is already created, use the following SCL:         ALTER TABLE Persons         OFAPT FAILE         The Column of the Column when the table is already created, use the following SCL:         ALTER TABLE Persons         Column of the Column when the table is already created, use the following SCL:         ALTER TABLE Persons         Column of the Column when the table is already created, use the following SCL:         ALTER TABLE Persons         Column of the Column when the table is already created, use the following SCL:         ALTER TABLE Persons         Column of the Column when the table is already created, use the following SCL:         ALTER TABLE Persons         Column of the Column when the table is already created, use the following SCL:         ALTER TABLE Persons         Column of the Column when the 'Declare's table is created:         OFAPCHARMY KEY constraint, use the following SCL:         ALTER TABLE Persons         Column of the SCH COLUMN (KEY Column)         Column when the 'Declare's table is created:         Create FARENCE SPecific KEY (Personal)         Column when the 'Declare's table is already created, use the following         SCL:         ALTER TABLE College         Create FARENCES Personal/Personal);         Column when the 'Declare's table is already created, use the following         SCL:         ALTER TABLE College         Create FARENCES Personal/Personal);         Column when the 'Declare's table is already created, use the following         SCL:         ALTE						
SQL PRIMARY KEY on CREAT TABLE  • The following SQL creates a FRIMARY KEY on the "D" column when the "Person" table is created:	<ul> <li>Primary keys must co</li> </ul>					
The following SQL creates a PRIMARY KEY on the 10° column when the "Denson" table is created: CEACE TABLE Persons CEACE TABLE Persons Denson PRIMARY KEY Constraint Denson PRIMARY KEY constraint on the "D" column when the table is already created, use the following SQL: ALTER TABLE Persons Dorouge a PRIMARY KEY constraint, use the following SQL: ALTER TABLE Persons Denson PRIMARY KEY constraint on the "D" column when the table is already created, use the following SQL: ALTER TABLE Persons Denson PRIMARY KEY constraint The FOREIGN KEY constraint Dorouge a PRIMARY KEY constraint, use the following SQL: ALTER TABLE Persons DROP PERMARY KEY constraint The FOREIGN KEY constraint Denson PRIMARY KEY constraint The FOREIGN KEY constraint DENSON KEY constraint DENSON KEY constraint The FOREIGN KEY constraint DENSON KEY constraint DENSON KEY constraint Constraint PRIMARY KEY constraint, use the following SQL: ALTER TABLE Denson COnstraint PRIMARY KEY constraint Constraint PRIMARY KEY constraint Constraint PRIMARY KEY constraint DENSON KEY Constraint The DERIEN KEY Constraint Constraint PRIMARY KEY constraint The DERIEN KEY Constraint Constraint A straint and the following SQL: ALTER TABLE Denson CONSTRAINT AND CONSTRAINT A			ary key can consist of single or multiple o	columns.		
<pre>Unit NOT NULL, LashNew surchur(256), Applied Directed PRIMARY KEY on ALTER TABLE To create a PRIMARY KEY constraint on the 10D column when the table is already created, use the following SQL: ALTER TABLE Persons ACTER TABLE Persons (Constraint) Constraint use the following SQL: ALTER TABLE Persons COREALE Constraint to use the following SQL: ALTER TABLE Constraint CoreAle and POREION KEY constraint on the "PersonDP" column when the "Orders" table is already created, use the following SQL: ALTER TABLE Constraint To compare the PersonDP, DEFERENCES Persons(PersonD); DROP a FOREION KEY constraint on the "PersonDP column when the "Orders" table is already created, use the following SQL: ALTER TABLE Constraint To compare the persons (Constraint on the the following SQL: ALTER TABLE Constraint To compare the persons (Constraint on the the following SQL: ALTER TABLE Constraint To compare the persons (Constraint on the the following SQL: ALTER TABLE Constraint To compare the persons (Constraint on the the following SQL: ALTER TABLE Constraint To compare the persons (Constraint, use the following SQL: ALTER TABLE Persons (Constraint), and the the following SQL: ALTER TABLE Persons (Constraint), and the following SQL: ALTER TABLE Persons (Constraint), and the following SQL: ALTER TABLE Persons (ALTER CONSTRAINT) CONSTRAINT Constraint (Use the following SQL: ALTER TABLE Persons (Constraint), and the following SQL: ALTER TABLE Persons (Constrai</pre>	The following SQL cr	ates a PRIMARY KEY on the "ID" column w	hen the "Persons" table is created:			
LatIName varbur(256) NOT NULL, FirstName varbur(256), Age int, Will of KeY (10): SSL PRIMARY KEY constant on the 'D' column when the table is already created, use the following SQL: ALTER TABLE Persons ADD PRIMARY KEY Constraint To drop a PRIMARY KEY constraint, use the following SQL: ALTER TABLE Persons RCPD PRIMARY KEY constraint is use the following SQL: ALTER TABLE Persons RCPD PRIMARY KEY Constraint To drop a PRIMARY KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY a failed in one table, that refers to the PERSONAL CV, an other table. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to read of the "PersonalD" column when the "Orders" table is already created, use the following SQL: ALTER TABLE Orders ADD FOREION KEY (PrionDID) REFERENCES Personal/PersonID); SQL FOREION KEY constraint To corpore of CMELCON KEY (PrionDID) REFERENCES Personal/PersonID); SQL FOREION KEY constraint, use the following SQL: ALTER TABLE Orders ADD FOREION KEY (PrionDID) REFERENCES Personal/PersonID); SQL DEFAULT events ADD FOREION KEY (PrionDID) REFERENCES Personal/PersonID); SQL DEFAULT Constraint ADD FOREION KEY (PrionDID) REFERENCES Personal/PersonID); SQL DEFAULT events ADD FOREION KEY (Pr		,				
Age int PRIMARY KEY on ALTER TABLE To create a PRIMARY KEY constraint on the 'D' column when the table is already created, use the following SOL: ALTER HARLE Persons DROP PRIMARY KEY constraint, use the following SOL: ALTER TABLE Persons DROP PRIMARY KEY: ii) Foreign key- Prevents actions that would destry links between tables. • The FORDER/KEY constraint is used to prevent actions that would destry links between tables. • The FORDER/KEY constraint is used to prevent actions that would destry links between tables. • The FORDER/KEY constraint is used to prevent actions that would destry links between tables. • The FORDER/KEY constraint's used to prevent actions that would destry links between tables. • The FORDER/KEY constraint's used to prevent actions that would destry links between tables. • The FORDER/KEY constraint on the "Person!! Column when the 'Ordens' table is created CREATE TABLE Ordens • Order Dirit NOT NULL, Person!! Constraint on the "Person!! Column when the 'Ordens' table is already created, use the following SQL: ALTER TABLE Ordens • ADD PORE/ENK KEY constraint on the "Person!! Column when the 'Ordens' table is already created, use the following SQL: ALTER TABLE Ordens • ADD PORE/ENK KEY constraint, use the following SQL: ALTER TABLE Ordens • DOROF FORE/ENK KEY constraint, use the following SQL: ALTER TABLE Ordens • DOROF FORE/ENK KEY constraint, use the following SQL: ALTER TABLE Ordens • DOROF FORE/ENK KEY constraint, use the following SQL: ALTER CONSTRAINT, walk is down is specified. • To the CREATE TABLE • To orden a DFFAULT request to action on the "Chy column when the "Person" table is created: • DOROF FORE/ENK KEY constraint, use the following SQL: ALTER CONSTRAINT, walk is that walk is a domain as adomnately assigned. <b>SQL EFAULT on ALTER TABLE</b> • To orden a DFFAULT constraint, use the following SQL: ALTER CONSTRAINT SUCCENTRAINT SUCCENT						
FRIMARY KEY (DI)):           Derived a FRIMARY KEY constraint on the 1D° column when the table is already created, use the following SQL:           ALTER TABLE Persons           DROP PRIMARY KEY constraint           To drop a PRIMARY KEY;           ii) Fordigh key- Prevents actions that would dectory links between tables.           • AFCRECON KEY;           iii) Fordigh key- Prevents actions that would dectory links between tables.           • The following SQL creates at DROEGN KEY or constraint, use the following SQL:           • AFCRECON KEY;           iii) Fordigh key- Prevents actions that would dectory links between tables.           • The following SQL creates at DROEGN KEY or constraint, use the PassotD* column when the "Orders" table is created.           • OrderD and TOT NULL.           OrderID and NOT NULL.           TRE TRABLE Orders           DROP FOREID						
SQL PRIMARY KEY on ALTER TABLE To create a FRMARY KEY constraint on the "D" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER TABLE Persons DROP PRIMARY KEY constraint, use the following SQL: ALTER TABLE Persons DROP PRIMARY KEY (D) CRO'S PRIMARY KEY (D) THE TABLE Persons ALTER TABLE PERSON ALTER TABLE PERSON ALTER TABLE PERSON ALTER TABLE POLICIES PERSON KEY or ALTER TABLE Order Universe ALTER TABLE POLICIES PERSON KEY ON ALTER TABLE TO TOTAL PERSON ALTER TABLE POLICIES ALTER TABLE						
ALTER TABLE Presons         DROP PRIMARY KEY constraint         To drop a PRIMARY KEY constraint.         To drop a PRIMARY KEY constraint.         To drop a PRIMARY KEY constraint.         IF Set PRIMARY KEY constraint.         IF Set PRIMARY KEY constraint.         IF Set PRIMARY KEY.         OrderDivine For NetPrimer and DY Set PRIMARY KEY.         Sol IF Set PRIMARY KEY.         Sol IF Set PRIMARY KEY.         Sol IF Set PRIMARY KEY.         IF Set PRIMARY KEY.         IF Set PRIMARY KEY.         Sol IF Set PRIMARY KEY.         Sol IF Set PRIMARY KEY.         Sol IF Set PRIMARY KEY.         IF Set PRIMARY KEY.         IF Set PRIMARY KEY. <td< td=""><td></td><td></td><td></td><td></td><td></td></td<>						
ADD FRIMARY KEY (10):       The MORP a FOMARY KEY constraint       The following SQL:         ALTER TABLE Persons       The Foreicrowite addross that would destry links between tables.       Image: Constraint on the set of the SQL (10) (10) (10) (10) (10) (10) (10) (10)		straint on the "ID" column when the table is	already created, use the following SQL:			
To drop a PRIMARY KEY constraint, use the following SQL: AFE TABLE Presenses DROP PRIMARY KEY: ii) Foreign key Prevents actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint is used to prevent actions that would destroy links between tables. • The FOREION KEY constraint on the TOT NULL. OrderNote in NOT NULL. Person10 Int, PRIMARY KEY (OrderD), PRIMARY KEY CONSTRAINT, USE the following SQL: ALTER TABLE PROFONE • To create a DEFAULT to value the following SQL: ALTER TABLE PROFONE • To create a DEFAULT Vey for few CQP, column when the "Presons" table is created. CERET TABLE PROFONE • To create a DEFAULT to reating the CQP, column when the table is already created, use the following SQL: ALTER TABLE PROFONE • To create a DEFAULT Constraint on the "CIP, column when the table is already created, use the following SQL: ALTER TABLE PROFON • The UNIQUE Constraint on the following SQL: ALTER TABLE PROFON • The UNIQUE Constraint on the following SQL: ALTER TABLE PROFON • The UNIQUE Constraint on the followin	ADD PRIMARY KEY (ID);					
ALTER TABLE Orders Provide Standard Control (Control (Con						
<pre>iii) Proreign key:- Prevents actions that would destroy links between tables. • The FOREIGN KEY on startint is used to prevent actions that would destroy links between tables. • A FOREIGN KEY a fait on to table, that refers to the [EMBAKY KEY] in another table. • A FOREIGN KEY a fait on the table, that refers to the [EMBAKY KEY] in another table. • A FOREIGN KEY a fait on the table, that refers to the [EMBAKY KEY] in another table. • A FOREIGN KEY a fait on the table, that refers to the [EMBAKY KEY] in another table. • CREATE TABLE Orders • ADD FOREIGN KEY constraint on the "PresonD" column when the "Orders" table is already created, use the following SOL: • ATTER TABLE Orders • ADD FOREIGN KEY (constraint on the "PresonD" column when the "Orders" table is already created, use the following SOL: • ATTER TABLE Orders • ADD FOREIGN KEY constraint, use the following SOL: • ATTER TABLE Orders • DROP FOREIGN KEY constraint, use the following SOL: • ATTER TABLE Orders • DROP FOREIGN KEY constraint, use the following SOL: • ATTER TABLE Orders • DROP FOREIGN KEY Constraint, use the following SOL: • ATTER TABLE Orders • DROP FOREIGN KEY Constraint weeth ord a discult value for a column. When no value is specified for the column during an INSERT • operation. the default value for a column the "Persons" table is created: • (LearName variant/255), • (A) per HUL Constraint • To create a DEFAULT Value for the "City" column when the "Persons" table is created: • CREATE TABLE Persons • (CREATE TABLE Persons) • (CREATE TABLE PERSONS) •</pre>		Istraint, use the following SQL.				
<ul> <li>The FOREIGN KEY is defined in one table, that refers to the ENARCY LCX in another table.</li> <li>A FOREIGN KEY is defined in one table, that refers to the ENARCY LCX in another table.</li> <li>The following SQL creates a FOREIGN KEY on the "PersonID" column when the "Orders" table is created: CREATE TABLE Orders         (OrderNumber in NOT NULL, PersonID int, PRIMARY KEY (OrderID), FOREIGN KEY on THE "PersonID" column when the "Orders" table is already created, use the following SQL: ALTER TABLE Orders AD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);</li> <li>SOL FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);</li> <li>BROM FOREIGN KEY (PersonID) REFERENCES Persons" table is specified for the column during an INSERT operation, the default value for a column if no value is specified.</li> <li>         The DEFAULT Netword is used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is a unotanically assigned.</li> <li>SOL DEFAULT on CREATE TABLE</li> <li>The tollowing SQL is a DEFAULT value for the City' column when the "Persons" table is created:</li></ul>						
<ul> <li>The FOREIGN KEY to activation the used to prevent actions that would destry links between tables.</li> <li>A FOREIGN KEY is a feel in one table, har dreves to the EMBARY KEY (another table.</li> <li>The following SQL creates a FOREIGN KEY on the "PersonID" column when the "Orders" table is created: CREATE TABLE Orders ( Orderful in NOT NULL, PRIMARY KEY (Conterl), FOREIGN KEY on ALTER TABLE To create a FOREIGN KEY (PersonID) REFERENCES Persons(PersonID); SQL FOREIGN KEY on ALTER TABLE To create a FOREIGN KEY (PersonID) REFERENCES Persons(PersonID); SQL FOREIGN KEY (PersonID) REFERENCES Persons(PersonID); SQL FOREIGN KEY (PersonID) REFERENCES Persons(PersonID); SQL TER TABLE Orders AD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID); BORD FOREIGN KEY (PersonID) REFERENCES Persons table is created:</li></ul>	iii) Foreign key:- Prevents actions	at would destroy links between tables.				
<ul> <li>The following SQL creates a FOREIGN KEY on the "PersonID" column when the "Orders" table is created: OREATE TABLE Orders ( Order/Order in NOT NULL, PERSONID in, PERSONID in, PERSONIC in, ALTER TABLE Persons ( PERSONIC in, ALTER TABLE Persons ( PERSONIC in, ALTER TABLE PERSONS ( ALTER TABLE PERSONS (</li></ul>	The FOREIGN KEY of the FORE	instraint is used to prevent actions that wou				
CREATE TABLE Orders ( Order10 int NOT NULL, Order10 int NOT NULL, Person10 int, PRIMARY KEY (Order10), FOREIGN KEY on ALTER TABLE To create a FOREIGN KEY Constraint on the "Person10" column when the "Orders" table is already created, use the following SQL: ALTER TABLE Orders ADD FOREIGN KEY Constraint on the "Person10" column when the "Orders" table is already created, use the following SQL: ALTER TABLE Orders ADD FOREIGN KEY Constraint To drop a FOREIGN KEY constraint, use the following SQL: ALTER TABLE Orders DROP FOREIGN KEY Constraint To drop a FOREIGN KEY Constraint (and the "Chy" column when the "Drosons" table is specified for the column during an INSERT operation, the default value is a subdictional yassigned. SQL DEFAULT value for a column if no value is specified. The DEFAULT value for a column if no value is specified. SQL DEFAULT value for a column if no value is specified. CREATE TABLE Persons (I) Unit NOT NULL, FristMane varcha(255) NOT NULL, FristMane varcha(255) DEFAULT "Sandnes"); SQL DEFAULT Constraint • To create a DEFAULT constraint on the "Chy" column when the "Dersons" table is created: CREATE TABLE Persons ALTER TABLE Persons ALTER TABLE Persons ALTER CRUE Constraint • To drop a DEFAULT Constraint, use the following SQL: ALTER TABLE Persons ALTER CRUE Constraint • To drop a DEFAULT Constraint, use the following SQL: ALTER TABLE Persons ALTER CRUE Constraint and the "Chy" column when the table is already created, use the following SQL: ALTER CRUE Constraint • To drop a DEFAULT Constraint, use the following SQL: ALTER TABLE Persons (UNIQUE constraint automaticality has a UNIQUE constraint automaticality has						
OrderNumber int NOT NULL, PersonDi Int, PersonDi Int, PERSONDI INT, PERSONDI REFERENCES Persons(PersonID);         SQL concellon KEY on ALTER TABLE         Image: Concentration of the "PersonID" column when the "Orders" table is already created, use the following SQL: ALTER TABLE Orders ADD FOREIGN KEY (personID) REFERENCES Persons(PersonID);         Image: Concentration of the "PersonID" column when the "Orders" table is already created, use the following SQL: ALTER TABLE Orders DROP a FOREIGN KEY constraint, use the following SQL: ALTER TABLE Orders DROP FOREIGN KEY constraint, use the following SQL: ALTER TABLE Orders DROP FOREIGN KEY constraint, use the following SQL: ALTER TABLE Orders DROP FOREIGN KEY constraint, use the following SQL: ALTER TABLE Orders DROP FOREIGN KEY constraint or due to set a default value for a column. When no value is specified for the column during an INSERT persons in the dista value is a unomatically assigned.         Image: Concentration of the column during an INSERT persons ( ID int NOT NULL, LashName varchar255), Age int, ORV varchar255), CREATE TABLE Persons ( ID int NOT NULL, FirstName varchar255), DT NULL, FirstName varchar255), Age int, CREATE TABLE Persons ( ID int NOT NULL, FirstName varchar255), Age int, CREATE TABLE Persons ALTER TABLE Persons ALTER TABLE Persons ALTER TABLE Persons ALTER TABLE Persons ALTER CRY, DEFAULT constraint on the "Chy" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER CRY, DEFAULT constraint and at different.         Image: Add Alter						
PersonID int. PRIMARY VEY (OrderID). PROPEICIN KEY (PersonID) REFERENCES Persons(PersonID)); SQL FOREIGN KEY on ALTER TABLE To create a POREIGN KEY constraint on the "PersonID" column when the "Orders" table is already created, use the following SQL: ALTER TABLE Orders ADD FOREIGN KEY constraint on the "PersonID" column when the "Orders" table is already created, use the following SQL: ALTER TABLE Orders ADD FOREIGN KEY constraint on the "DersonID" column when the "Orders" table is already created, use the following SQL: ALTER TABLE Orders DROP FOREIGN KEY constraint on the value is specified. • The DEFAULT Keywords is used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is automatically assigned. SQL DEFAULT on CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255), Age int, City varchar(255) DEFAULT Sendnes'): SQL DEFAULT constraint on the "City" column when the "Persons" table is created: ALTER TABLE Persons ( ID int NOT NULL, LastName varchar(255), Age int, City varchar(255) DEFAULT Sendnes'): SQL DEFAULT Constraint on the "City" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER City DEFAULT Constraint, use the following SQL: ALTER TABLE Persons ALTER City DEFAULT Constraint, use the following SQL: ALTER TABLE Persons ALTER City DEFAULT Constraint and and different. • The UNOUE constraint automatically has a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ALTER City DEFAULT Constraint automatically has a UNIQUE constraint automatically has a UNIQUE constraint automatically has a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ALTER City DEFAULT Constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID Int NOT NULL, LastName varchar(255), Age int, UNIQUE constraint on ALTER TABLE • To otere a UNIQUE constraint on the						
POREIGN KEY on ALTER TABLE       To create a FOREIGN KEY constraint on the "PersonID" column when the "Orders" table is already created, use the following SQL:       ALTER TABLE Orders       ALTER TABLE Orders       ADD FOREIGN KEY constraint       To drag a FOREIGN KEY constraint, use the following SQL:       ALTER TABLE Orders       ALTER TABLE Orders       To drag a FOREIGN KEY constraint, use the following SQL:       ALTER TABLE Orders       To drag a FOREIGN KEY constraint on the "order or a column. When no value is specified for the column during an INSERT       Operation. The default value is a udoratically assigned.         SQL DEFAULT Reyword is used to set a default value for a column. When no value is specified for the column during an INSERT       Operation. The default value is a udoratically assigned.         SQL DEFAULT regressions (       (ID in NOT NULL, LEST TABLE Persons (       (ID in NOT AUTER TABLE Persons (       (ID in NOT a DEFAULT Constraint on the "Orly" column when the table is already created, use the following SQL: ALTER City DOP DEFAULT constraint on the "Orly" column are different.       (ID in NOT NULL, FUNCTION SUGLE constraint on the "Orly" column are different.       (ID in NOT NULL, Constraint ano column are different.       (ID						
SQL FOREIGN KEY on ALTER TABLE       To create a FOREIGN KEY constraint on the "PersonID" column when the "Orders" table is already created, use the following SQL:       ALTER TABLE Orders       ADD FOREIGN KEY (DersonID) REFERENCES Persons(PersonID);         DROP a FOREIGN KEY constraint, use the following SQL:       ALTER TABLE Orders       DDOP FOREIGN KEY constraint, use the following SQL:         ALTER TABLE Orders       DDOP FOREIGN KEY constraint, use the following SQL:       ALTER TABLE Orders         DROP a FOREIGN KEY constraint, use the following SQL:       ALTER TABLE Orders         DROP FOREIGN KEY word is used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is automatically assigned.         SQL DEFAULT require for the "City" column when the "Persons" table is created:       CREATE TABLE Persons (         ID int NOT NULL,       Leasthame vanchar(255) NOT NULL,         FristName vanchar(255) NOT NULL,       FristName vanchar(255) NOT NULL,         City varchar(255) DEFAULT 'Sandnes');       Cut CREATE TABLE Persons         ALTER City SET DEFAULT constraint on the "City" column when the table is already created, use the following SQL:         ALTER City SET DEFAULT constraint, use the following SQL:         ALTER City SET DEFAULT constraint, use the following SQL:         ALTER City SET DEFAULT constraint, use the following SQL:         ALTER City SET DEFAULT constraint, use the following SQL:         ALTER City SET DEFAULT constr		PRIMARY KEY (OrderID),				
To create a FOREIGN KEY constraint on the "PersonID" column when the 'Orders' table is already created, use the following SQL: ALTER TABLE Orders ADD FOREIGN KEY Constraint To drop a FOREIGN KEY constraint, use the following SQL: ALTER TABLE Orders DROP FOREIGN KEY constraint, use the following SQL: ALTER TABLE Orders DROP FOREIGN KEY constraint, use the following SQL: ALTER TABLE Persons CREATE TABLE Persons () DID IN TOT NULL, LastName varchar(255), DEFAULT 'Sandnes'); SQL DEFAULT Constraint, use the following SQL: ALTER TABLE Persons ALT	SQL FOREIGN KEY on ALTER TA		NOES Persons(PersonID));			
ALTER TABLE Orders ADD FOREIGN KEY (ensemble) REFERENCES Persons(PersonID); DROP a FOREIGN KEY Constraint To drop a FOREIGN KEY Constraint, use the following SQL: ALTER TABLE Orders DROP FOREIGN KEY FK_PersonOrder; w) Default:- Sets a default value for a column if no value is specified. • The DEFAULT Keyword is used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is automatically assigned. SQL DEFAULT on CREATE TABLE The following SQL sets a DEFAULT value for the "Chy" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255), NOT NULL, FirstName varchar(255), DEFAULT 'Sandnes'); SQL DEFAULT on CREATE TABLE • To create a DEFAULT constraint on the "Chy" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER Chy SET DEFAULT constraint, use the following SQL: ALTER TABLE Persons ALTER Chy DROP DEFAULT; w)Unique:-Ensures that all values in a column are different. • The UNIQUE constraint ensures that all values in a column are different. • APRIMARY KEY constraint sprovide a guarantee for uniqueness for a column or set of columns. • APRIMARY KEY constraint per table, but ony one PRIMARY KEY constraint per table, but ony one PRIMARY KEY constraint per table. • The Volking SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( Dint NOT NULL, IsatName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, FirstName varchar(255),	To create a FOREIGN		hen the "Orders" table is already created	d, use the following		
ADD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);       10       CC         DROP a FOREIGN KEY constraint, use the following SQL:       ALTER TABLE Orders       10       CC         w) Default: Sets a default value for a column if no value is specified.       10       CC         • The DEFAULT reverse value is submatically assigned.       SQL DEFAULT value for a column if no value is specified.       10       CC         • The DEFAULT reverse value is automatically assigned.       SQL DEFAULT value for the "City" column when the "Persons" table is created:       CREATE TABLE Persons (       IIII IN TON TULL,       FirstName varchar(255), DEFAULT value for the "City" column when the "Persons" table is created;       CREATE TABLE Persons (       IIIII IN TON TULL,       FirstNam varchar(255), DEFAULT value for Column when the table is already created, use the following SQL:       ALTER TABLE Persons       ALTER TABLE Persons       ALTER City SET DEFAULT "Sandnes");         SQL DEFAULT constraint on the "City" column when the table is already created, use the following SQL:       ALTER TABLE Persons       ALTER City SET DEFAULT "Sandnes";       IIII IN OT NULL,       FirstNam varchar(255), DEFAULT "Sandnes";       IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII		Irders				
To drop a FOREIGN KEY constraint, use the following SQL: ALTER TABLE Orders DROP FOREIGN KEY FK_PersonOrder; v) Default: Sets a default value for a column if no value is specified for the column during an INSERT operation, the default value for a default value for a column. When no value is specified for the column during an INSERT operation, the default value for the 'City' column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255), Age int, City varchar(255) DEFAULT 'Sandnes'); SQL DEFAULT constraint on the "City" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER City SET DEFAULT Sandnes'; DROP a DEFAULT Constraint on the "City" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER City SET DEFAULT Sandnes'; DROP a DEFAULT Constraint on the "City" column are different. • To drop a DEFAULT Constraint, use the following SQL: ALTER City SET DEFAULT Sandnes; DROP a DEFAULT Constraint automatically has a UNIQUE constraints provide a guarantee for uniqueness for a column or set of columns. • A PRIMARY KEY constraint automatically has a UNIQUE constraints per table. But only one PRIMARY KEY constraint automatically has a UNIQUE			PersonID);			
ALTER TABLE Orders DROP FOREIGN KEY FK_PersonOrder; w) Default: Sets a default value for a column if no value is specified. The DEFAULT exported it value is a used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is a used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is a used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is a used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is a used to set a default value for a column when the "Persons" table is created: CREATE TABLE Persons ( I D int NOT NULL, LastName varchar(255), DEFAULT Sandnes'; SOL DEFAULT on ALTER TABLE • To create a DEFAULT constraint on the "City" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER TABLE Persons ALTER City SET DEFAULT Constraint, use the following SQL: ALTER TABLE Persons ALTER City DOP DEFAULT; vplurique: Ensures that all values in a column are different. • To the UNQUE constraint ensures that all values in a column are different. • Hou NUQUE constraint ensures that all values in a column are different. • Hou NUQUE constraint ensures that all values in a column are different. • Hou NUQUE constraint ensures that all values in a column are different. • Hou NUQUE constraint ensures that all values in a column are different. • Hou NUQUE constraint ensures that all values in a column are different. • Hou NUQUE constraint ensures that all values in a column are different. • Hou NUQUE constraint on the "D" column when the "Persons" table is created: CREATE TABLE Persons ( D Int NT NULL, EastName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, First				10	CO	
<pre>DROP FOREIGN KEY FK_PersonOrder; w) Default: Sets a default value for a column in ovalue is specified. • The DEFAULT Keyword is used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is automatically assigned. SOL DEFAULT on CREATE TABLE The following SQL sets a DEFAULT value for the "City" column when the "Persons" table is created:</pre>						
<ul> <li>The DEFAULT keyword is used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is automatically assigned.</li> <li>SQL DEFAULT on CREATE TABLE The following SQL sets a DEFAULT value for the "City" column when the "Persons" table is created:</li></ul>						
<ul> <li>The DEFAULT keyword is used to set a default value for a column. When no value is specified for the column during an INSERT operation, the default value is automatically assigned.</li> <li>SQL DEFAULT on CREATE TABLE The following SQL sets a DEFAULT value for the "City" column when the "Persons" table is created:</li></ul>						
operation, the default value is automatically assigned. SQL DEFAULT on CREATE TABLE The following SQL sets a DEFAULT value for the "City" column when the "Persons" table is created: CREATE TABLE Persons ( I to int NOT NULL, LastName varchar(255) NOT NULL, FristName varchar(255), Age int, City varchar(255) DEFAULT 'Sandnes'; SQL DEFAULT on ALTER TABLE To create a DEFAULT constraint on the "City" column when the table is already created, use the following SQL: ALTER City SET DEFAULT 'Sandnes'; DROP a DEFAULT constraint on the "City" column when the table is already created, use the following SQL: ALTER City SET DEFAULT 'Sandnes'; DROP a DEFAULT constraint, use the following SQL: ALTER City SET DEFAULT 'Sandnes'; VIUnique: Ensures that all values in a column are different.  • The UNIQUE constraint ensures that all values in a column are different. • The UNIQUE constraint nearchariteriatins provide a guarantee for uniqueness for a column or set of columns. • APRIMARY KEY constraints per table, but only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, FristName varchar(255), Age int, UNIQUE (f)): SQL UNIQUE Constraint on the "ID" column when the table is already created, use the following SQL:			When no volue is second of the state			
SQL DEFAULT on CREATE TABLE         The following SQL sets a DEFAULT value for the "City" column when the "Persons" table is created:         CREATE TABLE Persons (         ID int NOT NULL,         LastName varchar(255),         Age int,         City varchar(255) DEFAULT 'Sandnes');         SQL DEFAULT on ALTER TABLE         • To create a DEFAULT constraint on the "City" column when the table is already created, use the following SQL:         ALTER TABLE Persons         Both he UNIQUE constraint neouristants provide a guarantee for uniqueness for a column or set of columns.         • APRIMARY KEY constraints per table, but only one PRIMARY KEY constraint per table.         The following SQL creates a UNIQUE constraint on			. when no value is specified for the colur	min during an INSERT		
CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255) NOT NULL, FirstName varchar(255), Age int, City varchar(255) DEFAULT 'Sandnes'); SQL DEFAULT on ALTER TABLE • To create a DEFAULT constraint on the "City" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER TABLE Persons ALTER City SET DEFAULT 'Sandnes'; DROP a DEFAULT Constraint • To drop a DEFAULT constraint, use the following SQL: ALTER TABLE Persons ALTER TABLE Persons ALTER TABLE Persons ALTER TABLE Persons ALTER TABLE Persons ALTER TABLE Persons ALTER City DROP DEFAULT; v)Unique: -Ensures that all values in a column are different. • The UNIQUE constraint ensures that all values in a column are different. • Both the UNIQUE constraint automatically has a UNIQUE constraint. • APRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns. • A PRIMARY KEY constraint provide a guarantee for uniqueness for a column or set of columns. • A PRIMARY KEY constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID intNOT NULL, LastName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, FirstName varchar(255), Age int, UNIQUE ((D)): SOL UNIQUE Constraint on ALTER TABLE • To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:	SQL DEFAULT on CREATE TABL					
<ul> <li>ID int NOT NULL, LastName varchar(255) NOT NULL, FirstName varchar(255), Age int, City varchar(255) DEFAULT 'Sandnes');</li> <li>SOL DEFAULT on ALTER TABLE</li> <li>To create a DEFAULT constraint on the "City" column when the table is already created, use the following SQL: ALTER City SET DEFAULT 'Sandnes';</li> <li>DROP a DEFAULT Constraint, use the following SQL: ALTER City SET DEFAULT 'Sandnes';</li> <li>DROP a DEFAULT Constraint, use the following SQL: ALTER City DROP DEFAULT;</li> <li>vijUnique:- Ensures that all values in a column are different.</li> <li>Both the UNIQUE constraint ensures that all values in a column are different.</li> <li>Both the UNIQUE constraint automatically has a UNIQUE constraint.</li> <li>However, many UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons</li> <li>CREATE TABLE Persons</li> <li>UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:</li> <li>CREATE TABLE Persons</li> <li>ALTER City DROP DEFAULT;</li> <li>ViJUnique:- Ensures that all values in a column are different.</li> <li>However, many UNIQUE constraints provide a guarantee for uniqueness for a column or set of columns.</li> <li>However, many UNIQUE constraint per table, but only one PRIMARY KEY constraint per table.</li> <li>However, many UNIQUE constraint on the "ID" column when the "Persons" table is created:</li> <li>CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL, FirstName varchar(255), NOT NULL,</li> <li>SOL UNIQUE (OID);</li> <li>SOL UNIQUE Constraint on ALTER TABLE</li> <li>To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:</li> </ul>	•		ons" table is created:			
FirstName varchar(255), Age int, City varchar(255) DEFAULT 'Sandnes'); SQL DEFAULT on ALTER TABLE • To create a DEFAULT constraint on the "City" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER City SET DEFAULT 'Sandnes'; DROP a DEFAULT Constraint • To drop a DEFAULT constraint, use the following SQL: ALTER TABLE Persons ALTER City DROP DEFAULT; v)Unique:- Ensures that all values in a column are different. • The UNIQUE constraint ensures that all values in a column are different. • The UNIQUE constraint sutomatically has a UNIQUE constraint. • APRIMARY KEY constraint sutomatically has a UNIQUE constraint. • However, many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255) NOT NULL, FirstName varchar(255), Age int, UNIQUE (iD)); SQL UNIQUE Constraint on ALTER TABLE • To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:						
Age int, City varchar(255) DEFAULT 'Sandnes'); SQL DEFAULT on ALTER TABLE • To create a DEFAULT constraint on the "City" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER City SET DEFAULT 'Sandnes'; DROP a DEFAULT Constraint • To drop a DEFAULT constraint, use the following SQL: ALTER TABLE Persons ALTER City DROP DEFAULT; vjUnique:- Ensures that all values in a column are different. • The UNIQUE constraint ensures that all values in a column are different. • Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns. • A PRIMARY KEY constraint automatically has a UNIQUE constraint. • However, many UNIQUE constraint per table, but only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255), NOT NULL, FirstName varchar(255), Age int, UNIQUE (iD)); SQL UNIQUE Constraint on the "ID" column when the table is already created, use the following SQL:						
City varchar(255) DEFAULT 'Sandnes'); SQL DEFAULT on ALTER TABLE • To create a DEFAULT constraint on the "City" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER City SET DEFAULT 'Sandnes'; DROP a DEFAULT Constraint • To drop a DEFAULT constraint, use the following SQL: ALTER TABLE Persons ALTER City DROP DEFAULT; v)Unique:- Ensures that all values in a column are different. • The UNIQUE constraint ensures that all values in a column are different. • Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns. • A PRIMARY KEY constraint automatically has a UNIQUE constraint. • However, many UNIQUE constraint automatically has a UNIQUE constraint. • However, many UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255), NOT NULL, FirstName varchar(255), Age int, UNIQUE (DI)); SQL UNIQUE Constraint on ALTER TABLE • To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:						
To create a DEFAULT constraint on the "City" column when the table is already created, use the following SQL: ALTER TABLE Persons ALTER City SET DEFAULT 'Sandnes'; DROP a DEFAULT Constraint To drop a DEFAULT constraint, use the following SQL: ALTER TABLE Persons ALTER City DROP DEFAULT; //Unique:- Ensures that all values in a column are different. The UNIQUE constraint ensures that all values in a column are different. The UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns. A PRIMARY KEY constraint automatically has a UNIQUE constraint. However, many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255), NOT NULL, FirstName varchar(255), Age int, UNIQUE Constraint on ALTER TABLE To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL: To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:	Cit					
ALTER TABLE Persons ALTER City SET DEFAULT 'Sandnes'; DROP a DEFAULT constraint • To drop a DEFAULT constraint, use the following SQL: ALTER TABLE Persons ALTER City DROP DEFAULT; v)Unique:-Ensures that all values in a column are different. • The UNIQUE constraint ensures that all values in a column are different. • The UNIQUE constraint ensures that all values in a column are different. • Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns. • A PRIMARY KEY constraint automatically has a UNIQUE constraint. • However, many UNIQUE constraint on the But only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255), NOT NULL, FirstName varchar(255), Age int, UNIQUE (ID)); SQL UNIQUE Constraint on ALTER TABLE • To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:		constraint on the "City" column when the tal	he is already created use the following G	SOL:		
DROP a DEFAULT Constraint       • To drop a DEFAULT constraint, use the following SQL:         ALTER TABLE Persons       ALTER City DROP DEFAULT;         v)Unique: - Ensures that all values in a column are different.       • The UNIQUE constraint ensures that all values in a column are different.         • The UNIQUE constraint ensures that all values in a column are different.       • Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.         • A PRIMARY KEY constraint automatically has a UNIQUE constraint.       • However, many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table.         The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created:       CREATE TABLE Persons (         ID int NOT NULL,       ID int NOT NULL,         LastName varchar(255),       Age int,         UNIQUE (ID));       SQL UNIQUE Constraint on the "ID" column when the table is already created, use the following SQL:			איז איז מויטמעץ טרבמנכע, עשל נוופ וטווטאוווט ז			
To drop a DEFAULT constraint, use the following SQL:     ALTER TABLE Persons     ALTER City DROP DEFAULT;      v)Unique:- Ensures that all values in a column are different.         The UNIQUE constraint ensures that all values in a column are different.         Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.         A PRIMARY KEY constraint automatically has a UNIQUE constraint.         However, many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons (         ID int NOT NULL,         LastName varchar(255), NOT NULL,         FirstName varchar(255),         Age int,         UNIQUE (DD));         SQL UNIQUE Constraint on the "ID" column when the table is already created, use the following SQL:         Concreate a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:		FAULT 'Sandnes';				
ALTER TABLE Persons ALTER City DROP DEFAULT; v)Unique:- Ensures that all values in a column are different. • The UNIQUE constraint ensures that all values in a column are different. • Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns. • A PRIMARY KEY constraint automatically has a UNIQUE constraint. • However, many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255), NOT NULL, FirstName varchar(255), Age int, UNIQUE (ID)); SQL UNIQUE Constraint on ALTER TABLE • To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:		onstraint, use the following SOL:				
<ul> <li>v)Unique:- Ensures that all values in a column are different.</li> <li>The UNIQUE constraint ensures that all values in a column are different.</li> <li>Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.</li> <li>A PRIMARY KEY constraint automatically has a UNIQUE constraint.</li> <li>However, many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table.</li> <li>The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created:</li> <li>CREATE TABLE Persons (</li> <li>ID int NOT NULL,</li> <li>LastName varchar(255), NOT NULL,</li> <li>FirstName varchar(255),</li> <li>Age int,</li> <li>UNIQUE (ID));</li> <li>SQL UNIQUE Constraint on the "ID" column when the table is already created, use the following SQL:</li> </ul>	ALTER TABLE Persons					
The UNIQUE constraint ensures that all values in a column are different.     Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.     A PRIMARY KEY constraint automatically has a UNIQUE constraint.     However, many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons (     ID int NOT NULL,     LastName varchar(255), NOT NULL,     FirstName varchar(255),     Age int,     UNIQUE (D)); SQL UNIQUE Constraint on the "ID" column when the table is already created, use the following SQL:	ALTER City DROP DEFA	LT;				
Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.     A PRIMARY KEY constraint automatically has a UNIQUE constraint.     However, many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255), NOT NULL, FirstName varchar(255), Age int, UNIQUE (ID)); SQL UNIQUE Constraint on the "ID" column when the table is already created, use the following SQL:						
A PRIMARY KEY constraint automatically has a UNIQUE constraint.     However, many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons (     ID int NOT NULL,     LastName varchar(255), NOT NULL,     FirstName varchar(255),     Age int,     UNIQUE (ID)); SQL UNIQUE Constraint on the "ID" column when the table is already created, use the following SQL:				of columns		
However, many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table. The following SQL creates a UNIQUE constraint on the "ID" column when the "Persons" table is created: CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255) NOT NULL, FirstName varchar(255), Age int, UNIQUE (ID)); SQL UNIQUE Constraint on ALTER TABLE     To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:				or columns.		
CREATE TABLE Persons ( ID int NOT NULL, LastName varchar(255) NOT NULL, FirstName varchar(255), Age int, UNIQUE (ID)); SQL UNIQUE Constraint on ALTER TABLE • To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:	<ul> <li>However, many UNIC</li> </ul>	JE constraints per table, but only one PRIM	ARY KEY constraint per table.			
ID int NOT NULL, LastName varchar(255) NOT NULL, FirstName varchar(255), Age int, UNIQUE (ID)); SQL UNIQUE Constraint on ALTER TABLE • To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:		E constraint on the "ID" column when the "F	Persons" table is created:			
FirstName varchar(255), Age int, UNIQUE (ID)); SQL UNIQUE Constraint on ALTER TABLE • To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:						
Age int, UNIQUE (ID)); SQL UNIQUE Constraint on ALTER TABLE • To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:		L,				
UNIQUE (ID)); SQL UNIQUE Constraint on ALTER TABLE  To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:						
To create a UNIQUE constraint on the "ID" column when the table is already created, use the following SQL:	UNIQUE (ID));					
			is already created use the following SO	I		
			is an easy or cated, use the following SQI	<u>-</u> .		
DROP a UNIQUE Constraint		D UNIQUE (ID);				

	ALTER TABLE Persons DROP INDEX UC_Person;			
	<ul> <li>Consider the following tables: works (Pname, Cname, Salary) lives (Pname, Street, City) located-In (Cname, City) write the following queries in SQL:</li> <li>i) List the names of the people who work for the company 'Wipro' along with the cities they live in.</li> <li>ii) Find the names of the persons who do not work for 'Infosys'.</li> <li>iii) Find the people whose salaries are more than that of all of the 'oracle' employees.</li> <li>iv) Find the persons who works and lives in the same city. (10 Marks)</li> </ul>			
	<ul> <li>i) List the names of the people who work for the company 'Wipro the cities they live in.</li> </ul>	o' along with		
	SELECT w.Pname, I.City FROM works w JOIN lives I ON w.Pn WHERE w.Cname = 'Wipro';	ame = I.Pname		
6	<sup>6</sup> ii) Find the names of the persons who do not work for 'Infosys'.		CO3	L3
	SELECT I.Pname FROM lives I WHERE I.Pname NOT IN ( SE w.Pname FROM works w WHERE w.Cname = 'Infosys' );	ELECT		
	iii) Find the people whose salaries are more than that of all of th employees.	e 'Oracle'		
	SELECT w1.Pname FROM works w1 WHERE w1.Salary > ALL ( SELECT w2.Salary FROM works w2 WHERE w2.Cname = 'Oracle' );			
	iv) Find the persons who work and live in the same city.			
	SELECT I.Pname FROM lives I JOIN located_In li ON I.City = li works w ON I.Pname = w.Pname AND w.Cname = li.Cname;	i.City JOIN		

CCI

HOD