DATA TYPES

DATA TYPES

DATA EXISTS IN VARIOUS FORMS

NUMBERS, TEXT, IMAGES AND VIDEOS

FINANCIAL DATA TYPICALLY EXISTS IN NUMERICAL FORM - BILL AMOUNT, BANK BALANCE, TRANSACTION AMOUNTS ETC

DATA TYPES

DATA EXISTS IN VARIOUS FORMS

NUMBERS, TEXT, IMAGES AND VIDEOS

ids	Rank	bday	Gender	Athlete
26582	.	.	Male	Athlete
46919	.	.	Female	Athlete
20230	Freshman	02-Jan-1996	Male	Athlete
21083	Freshman	22-Dec-1995	Female	Non-athlete
21939	Freshman	12-Dec-1995	Male	Athlete
23017	Freshman	29-Nov-1995	.	Non-athlete
23217	Freshman	26-Nov-1995	Male	Non-athlete
23889	Freshman	18-Nov-1995	Male	Athlete
24985	Freshman	05-Nov-1995	Female	Non-athlete

SOME INFORMATION IS INHERENTLY TEXTUAL, LIKE CATEGORIES, NAMES, ADDRESSES AND OTHER DESCRIPTIVE INFORMATION

DATA TYPES

DATA EXISTS IN VARIOUS FORMS

NUMBERS, TEXT, IMAGES AND VIDEOS

MUCH OF THE INFORMATION IN TODAY'S WORLD EXISTS IN FORM OF VIDEOS AND PICTURES

DATA TYPES

DATA EXISTS IN VARIOUS FORMS

NUMBERS, TEXT, IMAGES AND VIDEOS

WE CAN STORE AND PROCESS ALL THESE DIFFERENT TYPES OF DATA IN HIVE

DATA TYPES

HIVE SUPPORTS TWO MAJOR DATA TYPES

PRIMITIVE

COLLECTION

DATA TYPES

HIVE SUPPORTS TWO MAJOR DATA TYPES

PRIMITIVE

COLLECTION

PRIMITIVE DATA TYPE

INCLUDES BOOLEAN, NUMERIC, STRING AND TIMESTAMP TYPES

EACH OF THESE DATA TYPES MAPS TO A CORRESPONDING DATA TYPE IN JAVA

PRIMITIVE DATA TYPE

INCLUDES BOOLEAN, NUMERIC, STRING AND TIMESTAMP TYPES

BOOLEAN
(TRUE/FALSE)

```
CREATE TABLE Customers
(
CustomerID INT,
IsBlocked BOOLEAN,
IsPremiumCustomer BOOLEAN
);
```

BOOLEAN IS USED MOST OFTEN WHEN THE DATA FIELD ANSWERS A SIMPLE YES/NO QUESTION

PRIMITIVE DATA TYPE

INCLUDES BOOLEAN, NUMERIC, STRING AND TIMESTAMP TYPES

INTEGERS

DECIMALS

PRIMITIVE DATA TYPE

INCLUDES BOOLEAN, NUMERIC, STRING AND TIMESTAMP TYPES

INTEGERS

TINYINT
(1-BYTE)
-128 TO 127

SMALLINT
(2-BYTE)
-32,768 TO 32,768

INT
(4-BYTE)
$(-2^{15}$ TO $2^{15}-1)$

BIGINT
(8-BYTE)
$(-2^{63}$ TO $2^{63}-1)$

PRIMITIVE DATA TYPE INTEGERS

TINYINT
(1-BYTE)
-128 TO 127

SMALLINT
(2-BYTE)
-32,768 TO 32,768

INT
(4-BYTE)
$(-2^{15}$ TO $2^{15}-1)$

BIGINT
(8-BYTE)
$(-2^{63}$ TO $2^{63}-1)$

```
CREATE TABLE ECOMMERCE_VISITS
(
VisitID BIGINT,
CustomerID BIGINT,
PageID INT,
SessionID SMALLINT,
PRODUCTCATEGORY TINYINT
);
```

PRIMITIVE DATA TYPE INTEGERS

TINYINT	SMALLINT	INT	BIGINT
(1-BYTE)	(2-BYTE)	(4-BYTE)	(8-BYTE)
-128 TO 127	-32,768 TO 32,768	$(-2^{15}$ TO $2^{15}-1)$	$(-2^{63}$ TO $2^{63}-1)$

```
CREATE TABLE ECOMMERCE_VISITS
(
VisitID BIGINT,
CustomerID BIGINT,
PageID INT
SessionDURATION SMALLINT,
PRODUCTCATEGORY TINYINT
);
```

THESE VARIABLES CAN EASILY RUN INTO BILLIONS

PRIMITIVE DATA TYPE INTEGERS

TINYINT	SMALLINT	INT	BIGINT
(1-BYTE)	(2-BYTE)	(4-BYTE)	(8-BYTE)
-128 TO 127	-32,768 TO 32,768	$(-2^{15}$ TO $2^{15}-1)$	$(-2^{63}$ TO $2^{63}-1)$

```
CREATE TABLE ECOMMERCE_VISITS
(
VisitID BIGINT,
CustomerID BIGINT,
PageID INT,
SessionID SMALLINT,
PRODUCTCATEGORY TINYINT
);
```

NUMBER OF PAGES ON A SITE WILL BE LESS THAN A BILLION

PRIMITIVE DATA TYPE INTEGERS

TINYINT	SMALLINT	INT	BIGINT
(1-BYTE)	(2-BYTE)	(4-BYTE)	(8-BYTE)
-128 TO 127	-32,768 TO 32,768	$(-2^{15}$ TO $2^{15}-1)$	$(-2^{63}$ TO $2^{63}-1)$

```
CREATE TABLE ECOMMERCE_VISITS
(
VisitID BIGINT,
CustomerID BIGINT,
PageID INT,
SessionID SMALLINT,
PRODUCTCATEGORY TINYINT
);
```

NUMBER OF SESSIONS A CUSTOMER HAS IN A DAY, POSSIBLY NOT MORE THAN 32768

PRIMITIVE DATA TYPE INTEGERS

TINYINT	SMALLINT	INT	BIGINT
(1-BYTE)	(2-BYTE)	(4-BYTE)	(8-BYTE)
-128 TO 127	-32,768 TO 32,768	$(-2^{15}$ TO $2^{15}-1)$	$(-2^{63}$ TO $2^{63}-1)$

```
CREATE TABLE ECOMMERCE_VISITS
(
VisitID BIGINT,
CustomerID BIGINT,
PageID INT,
SessionID SMALLINT,
PRODUCTCATEGORY TINYINT
);
```

NUMBER OF PRODUCT CATEGORIES WILL PROBABLY BE LESS THAN 100

PRIMITIVE DATA TYPE

INCLUDES BOOLEAN, NUMERIC, STRING AND TIMESTAMP TYPES

INTEGERS

DECIMALS

PRIMITIVE DATA TYPE

INCLUDES BOOLEAN, NUMERIC, STRING AND TIMESTAMP TYPES

DECIMALS

FLOAT
(4-BYTE)

DOUBLE
(8-BYTE)

DECIMAL
(ARBITRARY
PRECISION
SIGNED NUMBER)

PRIMITIVE DATA TYPE DECIMALS

FLOAT
(4-BYTE)

DOUBLE
(8-BYTE)

```
CREATE TABLE EXPERIMENT_OBSERVATION
(
EXPERIMENTID BIGINT,
AVOGADRO_NUMBER FLOAT,
VALUE_OF_PI DOUBLE
);
```

PRIMITIVE DATA TYPE

DECIMALS

FLOAT
(4-BYTE)

DOUBLE
(8-BYTE)

```
CREATE TABLE EXPERIMENT_OBSERVATION
(
EXPERIMENTID BIGINT,
AVOGADRO_NUMBER FLOAT,
VALUE_OF_PI DOUBLE
);
```

DOUBLE HAS A HIGHER PRECISION AND RANGE AS COMPARED TO FLOAT

PRIMITIVE DATA TYPE

DECIMALS

DECIMAL
(ARBITRARY PRECISION SIGNED NUMBER)

```
CREATE TABLE ORDERS_TABLE
 (
ORDERID BIGINT,
ORDER_VALUE DECIMAL(10,2),
SELLING_PRICE (10)
);
```

PRIMITIVE DATA TYPE

DECIMALS

DECIMAL
(ARBITRARY PRECISION SIGNED NUMBER)

```
ORDER_VALUE DECIMAL(10,2)
```

10 IS THE TOTAL NUMBER OF DIGITS IN THIS FIELD, 2 IS THE NUMBER OF DECIMAL PLACES

PRIMITIVE DATA TYPE

DECIMAL
(ARBITRARY PRECISION SIGNED NUMBER)

```
CREATE TABLE ORDERS_TABLE
(
ORDERID BIGINT,
ORDER_VALUE DECIMAL(10,2),
SELLING_PRICE (10)
);
```

ORDER VALUE WILL VARY FROM
-99,999,999.99 TO 99,999,999.99

PRIMITIVE DATA TYPE

DECIMAL
(ARBITRARY PRECISION SIGNED NUMBER)

```
CREATE TABLE ORDERS_TABLE
 (
ORDERID BIGINT,
ORDER_VALUE DECIMAL(10,2),
SELLING_PRICE (10)
);
```

SELLING PRICE WILL VARY FROM
-9,999,999,999 TO 9,999,999,999

PRIMITIVE DATA TYPE

INCLUDES NUMERIC, BOOLEAN, STRING AND TIMESTAMP

STRING
(UNBOUNDED VARIABLE LENGTH CHARACTER STRING)

CHAR
(FIXED-LENGTH CHARACTER STRING)

VARCHAR
(VARIABLE-LENGTH CHARACTER STRING)

EXAMPLES LIKE "JOHN", 'M', 'F'

CHAR (FIXED-LENGTH CHARACTER STRING)

VS

VARCHAR (VARIABLE-LENGTH CHARACTER STRING)

1 CHARACTER TAKES 1 BYTE PER CHARACTER

VARCHAR(100) HOLDS THE VALUE 'ABC' OCCUPIES 5 BYTES

CHAR(100) HOLDS THE VALUE 'ABC' OCCUPIES 100 BYTES

VARCHAR SAVES SPACE

STRING
(UNBOUNDED
VARIABLE LENGTH
CHARACTER
STRING)

VS

VARCHAR
(VARIABLE-
LENGTH
CHARACTHER
STRING)

1 CHARACTER TAKES 1 BYTE PER CHARACTER

BY DEFAULT STRING IS MAPPED TO VARCHAR(32762)

VARCHAR SAVES SPACE

PRIMITIVE DATA TYPE

INCLUDES NUMERIC, BOOLEAN, STRING AND TIMESTAMP

TIMESTAMP
TIMESTAMP WITH
NANOSECOND PRECISION

'1970-01-01 00:00:01'

'2038-01-19 03:14:07'

DATE
DATE

'1970-01-01'

'2012-02-01'

DATA TYPES

HIVE SUPPORTS TWO MAJOR DATA TYPES

PRIMITIVE

COLLECTION

COLLECTION DATA TYPE

ARRAY

MAP

STRUCT

UNION

HIVEQL HAS 4 COLLECTION DATA TYPES

COLLECTION DATA TYPE

ARRAY

MAP

STRUCT

UNION

THESE DATA TYPES CAN BE USED TO EMBED LOTS OF INFORMATION INTO A SINGLE COLUMN

ARRAY COLLECTION DATA TYPE

AN ARRAY IS A COLLECTION OF VARIABLES ALL OF THE SAME TYPE

LET US ASSUME YOU HAVE A LIST OF FOODS THAT YOU LIKE

FOOD ITEMS
BURGER
FRIES
PIZZA
PASTA

ARRAY COLLECTION DATA TYPE

AN ARRAY IS A COLLECTION OF VARIABLES ALL OF THE SAME TYPE

FOOD ITEMS
BURGER
FRIES
PIZZA
PASTA

ALL OF THE ITEMS IN THE LIST HAVE SIMILAR PROPERTIES AND BELONG TO THE SAME GROUP I.E. JUNK FOOD

THIS IS AN ARRAY OF JUNK FOOD ITEMS

ARRAY COLLECTION DATA TYPE

AN ARRAY IS A COLLECTION OF VARIABLES ALL OF THE SAME TYPE

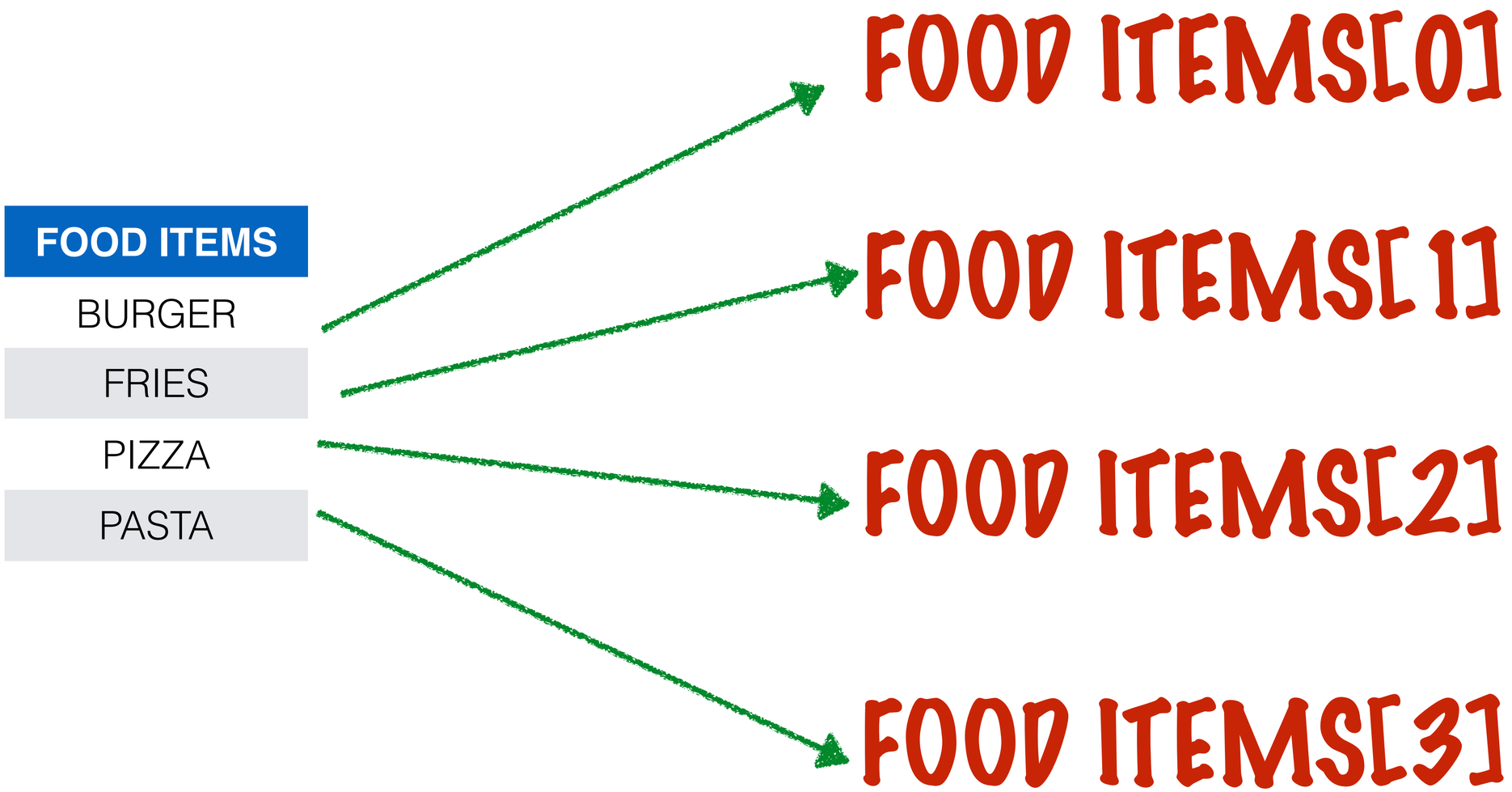
FOOD ITEMS
BURGER
FRIES
PIZZA
PASTA

THIS IS AN ARRAY OF JUNK FOOD ITEMS

USING ARRAYS, WE CAN ADDRESS A COLLECTION OF VARIABLES OF SAME TYPE AS A GROUP

USING ARRAYS, WE CAN ADDRESS A COLLECTION OF VARIABLES OF SAME TYPE AS A GROUP

AN ARRAY IS A COLLECTION OF VARIABLES ALL OF THE SAME TYPE



AN ARRAY IS A COLLECTION OF VARIABLES ALL OF THE SAME TYPE

ARRAYS HAVE A FIXED SIZE

ONLY ARRAYS OF PRIMITIVE DATA TYPES ARE ALLOWED IN HIVE

ARRAY

ARRAYS OF PRIMITIVE DATA TYPES CAN BE USED LIKE THIS

```
CREATE TABLE CUSTOMER_TABLE
(
CUSTOMER_ID BIGINT(100),
CUSTOMER_NAME VARCHAR(100),
ADDRESSES    ARRAY<VARCHAR(1000)>,
PURCHASED_PRODUCT_IDs ARRAY<BIGINT>
);
```

ARRAY

ALL THE COLLECTION DATA TYPES ARE SPECIFIED THE WAY JAVA GENERICS ARE

```
CREATE TABLE CUSTOMER_TABLE
(
CUSTOMER_ID BIGINT(100),
CUSTOMER_NAME VARCHAR(100),
ADDRESSES    ARRAY<VARCHAR(1000)>,
PURCHASED_PRODUCT_IDs ARRAY<BIGINT>
);
```

ARRAY

ARRAYS ARE GENERICS WITH 1 TYPE PARAMETER

```
CREATE TABLE CUSTOMER_TABLE
(
CUSTOMER_ID BIGINT(100),
CUSTOMER_NAME VARCHAR(100),
ADDRESSES    ARRAY<VARCHAR(1000)>,
PURCHASED_PRODUCT_IDs ARRAY<BIGINT>
);
```

MAP

SIMILAR TO THEIR NAMESAKES IN JAVA

THE MAP IS A DATA TYPE WHERE DATA IS REPRESENTED IN THE FORM OF PAIRS.

MAP IS REPRESENTED AS (KEY, VALUE) PAIR

EACH KEY IN MAP IS UNIQUE AND IS MAPPED TO A VALUE

MAP

MAP IS REPRESENTED AS (KEY, VALUE) PAIR

KEY AND VALUE CAN BE OF ANY DATA TYPE

A DICTIONARY IS AN EXAMPLE OF A MAP

{WORD, MEANING}

KEY

DATA TYPE =STRING

VALUE

DATA TYPE = ARRAY<VARCHAR>

MAP

{WORD, MEANING}

VALUE

DATA TYPE = ARRAY<VARCHAR>

MEANINGS ARE TYPICALLY SENTENCES MADE OF WORDS OF DIFFERENT LENGTH I.E. A LIST OF WORDS

MAP

{WORD, MEANING}

VALUE

DATA TYPE = ARRAY<VARCHAR>

THE VARCHAR IS FOR EACH WORD AND THE ARRAY REPRESENTS THE SENTENCE OF WORDS

MAP

MAP IS REPRESENTED AS (KEY, VALUE) PAIR

EACH KEY IN MAP IS MAPPED TO A VALUE

A DICTIONARY IS AN EXAMPLE OF A MAP

{WORD, MEANING}

```
CREATE TABLE OXFORD_DICTIONARY
(
PAGE_ID INT,
DICTIONARY MAP<STRING,ARRAY<VARCHAR(1000)>>
);
```

STRUCT

A STRUCTURE IS A DATA TYPE THAT IS USED TO REFER TO A LOGICAL GROUP OF VARIABLES

STRUCT

A STRUCTURE IS A DATA TYPE THAT IS USED TO REFER TO A LOGICAL GROUP OF VARIABLES

THE LOGICAL GROUP CAN HAVE THE SAME OR DIFFERENT DATA TYPES

STRUCT

A STRUCTURE IS A DATA TYPE THAT IS USED TO REFER TO A LOGICAL GROUP OF VARIABLES

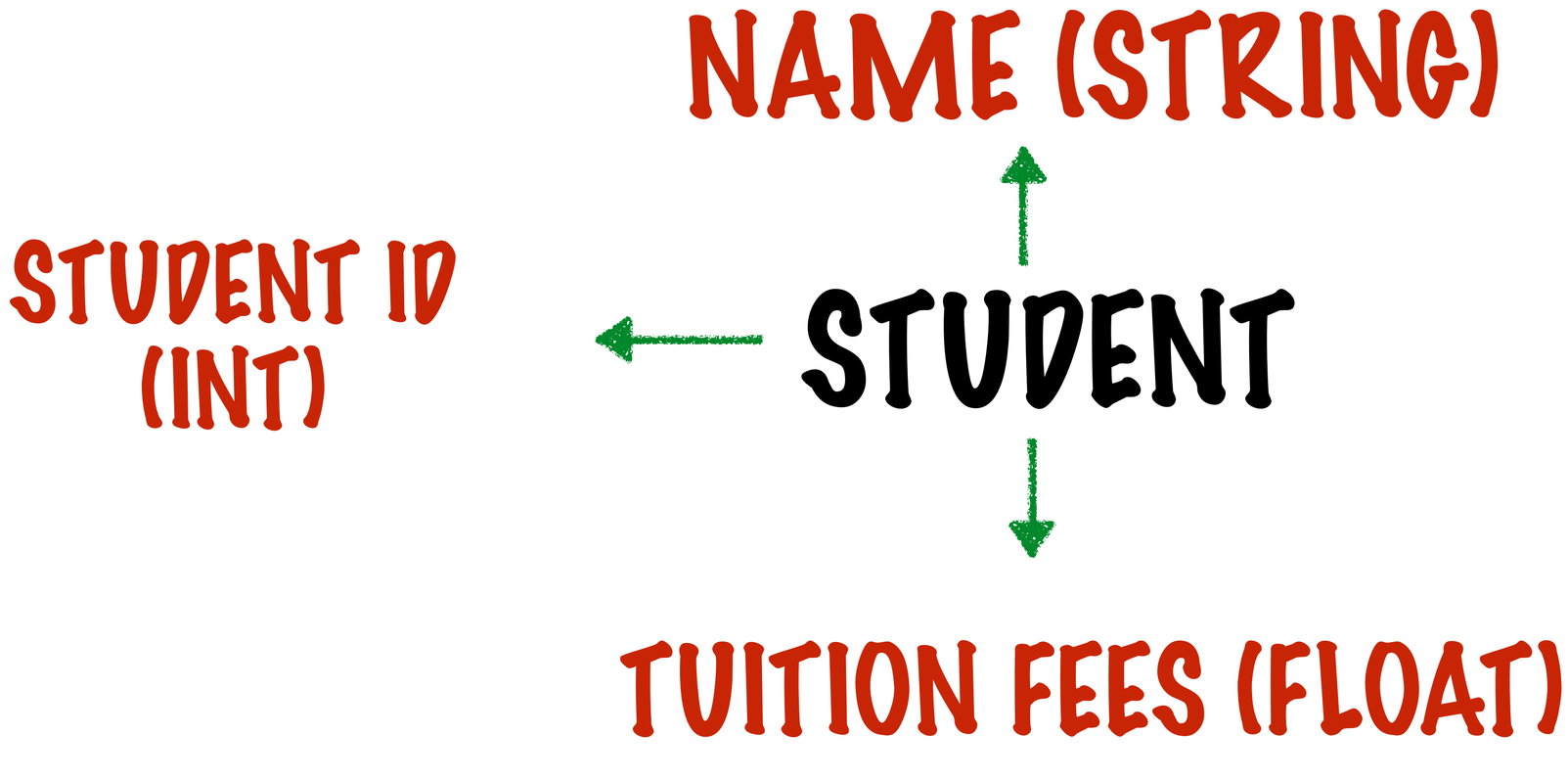
THE LOGICAL GROUP CAN HAVE THE SAME OR DIFFERENT DATA TYPES

THIS LOGICAL GROUP IS CALLED BY A SINGLE NAME

COMPLEX DATA TYPE

STRUCT

LET US SAY WE WANT TO STORE INFORMATION RELATED TO A STUDENT



COMPLEX DATA TYPE STRUCT

DIFFERENT VARIABLES DEFINE DIFFERENT ATTRIBUTES OF THE ENTITY STUDENT

STUDENT

{ NAME (STRING) ,
STUDENT ID (INT) ,
TUTION FEES (FLOAT)
}

AND ALL THE VARIABLES CAN BE ADDRESSED BY A SINGLE NAME STUDENT

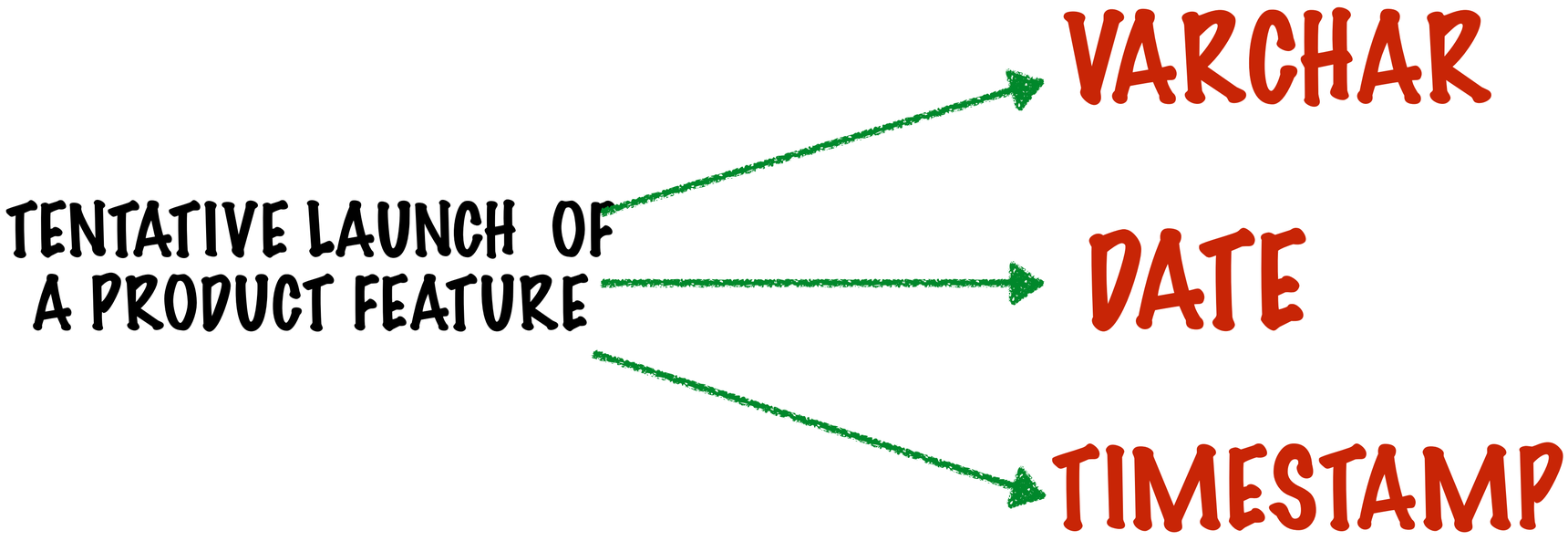
UNION

UNION IS A DATA TYPE THAT CAN HAVE ONE OF MANY DATA TYPES

UNION

UNION IS A DATA TYPE THAT CAN HAVE ONE OF MANY DATA TYPES

IT IS USEFUL WHEN YOU WANT TO STORE SOMETHING THAT COULD BE ONE OF SEVERAL TYPES



UNION

IT IS USEFUL WHEN YOU WANT TO STORE SOMETHING THAT COULD BE ONE OF SEVERAL TYPES

TENTATIVE LAUNCH OF A PRODUCT FEATURE

{ VARCHAR ,
DATE ,
TIMESTAMP }

THE UNION VARIABLE CAN STORE DATA IN ANY ONE OF THE THREE FORMATS

ONLY ONE OF THE FORMATS WILL BE APPLICABLE FOR ANY ONE RECORD

STRUCT

UNION

SYNTAX FOR STRUCT AND UNION WILL BE LIKE THIS

```
CREATE TABLE RANDOM_TABLE
(
CUSTOMER_ID BIGINT(100),
COLUMN_A STRUCT<a:STRING,b:INT,c:BOOLEAN>,
Column_B UNIONTYPE<STRING,INT,BOOLEAN>
);
```

OPERATORS AND FUNCTIONS

OPERATORS AND FUNCTIONS

HIVE HAS ALL THE IMPORTANT BUILT-IN OPERATORS

RELATIONAL OPERATORS

ARITHMETIC OPERATORS

LOGICAL OPERATORS

RELATIONAL OPERATORS

THESE OPERATORS ARE USED TO COMPARE TWO OPERANDS.
THEY RETURN BOOLEAN DATA TYPE (TRUE/FALSE)

A = B
A != B
A < B
A <= B
A > B
A >= B

ALL PRIMITIVE TYPES

IS NULL
IS NOT NULL

ALL TYPES

IS LIKE

STRING

RELATIONAL OPERATORS

THESE ARE USED TO COMPARE TWO OPERANDS.

THEY RETURN BOOLEAN DATA TYPE (TRUE/FALSE)

```
select customerID,OrderID
from OrderTable
where order_city_code = 'NY';
```

THIS QUERY WILL OUTPUT ONLY THOSE ORDERS WHICH HAVE BEEN PLACED FROM NEWYORK

ARITHMETIC OPERATORS

THESE SUPPORT ARITHMETIC OPERATIONS

A + B

A - B

A * B

A/B

A % B

BITWISE OPERATOR

A & B

A | B

A ^ B

LOGICAL OPERATORS

THEY RETURN EITHER TRUE OR FALSE

A AND B	A && B
A OR B	A || B
NOT B	!B

LOGICAL OPERATORS

```
select customerID,OrderID
from OrderTable
where order_city_code = 'NY'
AND PRODUCT_CATEGORY = 'Mobiles';
```

THIS QUERY WILL OUTPUT ONLY THOSE ORDERS WHICH HAVE BEEN PLACED FROM NEWYORK

AND ARE OF MOBILES

LOGICAL OPERATORS

```
select customerID,OrderID
from OrderTable
where order_city_code = 'NY'
AND NOT(PRODUCT_CATEGORY = 'Mobiles);
```

THIS QUERY WILL OUTPUT ONLY THOSE ORDERS WHICH HAVE BEEN PLACED FROM NEWYORK

AND ARE NOT MOBILES

LOGICAL OPERATORS

```
select customerID,OrderID
from OrderTable
where order_city_code = 'NY'
OR order_city_code = 'SF';
```

THIS QUERY WILL OUTPUT ONLY THOSE ORDERS WHICH HAVE BEEN PLACED FROM NEWYORK

OR SANFRANCISCO

OPERATORS AND FUNCTIONS

HIVE ALSO HAS A NUMBER OF BUILT-IN FUNCTIONS

MATHEMATICAL FUNCTIONS

COLLECTION FUNCTIONS

CONVERSION FUNCTIONS

TYPE FUNCTIONS

DATE FUNCTIONS

CONDITIONAL FUNCTIONS

STRING FUNCTIONS

OPERATORS AND FUNCTIONS

HIVE ALSO HAS A NUMBER OF BUILT-IN FUNCTIONS

WHAT MAKES HIVE VERY POWERFUL IS:

USER DEFINED FUNCTIONS

USERS ARE NOT LIMITED BY WHAT IS PRESENT
IN LIBRARIES, BUT CAN BUILD THEIR OWN

TABLES

TABLES

HIVE ARRANGES DATA IN FORM OF DATABASES AND TABLES

A TABLE IS A COLLECTION OF RELATED DATA HELD IN A STRUCTURED FORMAT WITHIN A DATABASE.

IT CONSISTS OF FIELDS (COLUMNS), AND ROWS.

TABLES

HIVE ARRANGES DATA IN FORM OF DATABASES AND TABLES

A DATABASE IS A COLLECTION OF TABLES

LET'S SAY WE HAVE A TABLE WITH STUDENT DATA

COLUMNS ARE NAMED 'STUDENTID', 'FIRSTNAME', 'LASTNAME', 'GENDER' AND 'EMAIL'

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh	M	navdeep@loonycorn.com
4	Vitthal	Srinivasan	M	vitthal@loonycorn.com

THIS IS A TABLE NAMED 'STUDENTS'

LET'S SAY WE HAVE A TABLE WITH STUDENT DATA

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh	M	navdeep@loonycorn.com
4	Vitthal	Srinivasan	M	vitthal@loonycorn.com

THIS IS A TABLE NAMED 'STUDENTS'

THE COLUMNS ARE NAMED 'STUDENT ID', 'FIRSTNAME', 'LASTNAME', 'GENDER' AND 'EMAIL'

THIS DATA SITS IN HDFS

HADOOP DISTRIBUTED FILE SYSTEM

THE DATA IS REFERRED TO AS HIVE DATA

TABLES IN HIVE HAVE TWO TYPES OF DATA

METADATA

THE HIVE METASTORE SERVICE STORES THE METADATA FOR HIVE TABLES AND PARTITIONS

HIVE DATA

THIS DATA SITS IN HDFS

HADOOP DISTRIBUTED FILE SYSTEM

METADATA

THE HIVE METASTORE SERVICE STORES THE METADATA FOR HIVE TABLES AND PARTITIONS

THIS DATA IS STORED IN A RELATIONAL DATABASE, AND PROVIDES CLIENTS (INCLUDING HIVE) ACCESS TO THIS INFORMATION VIA THE METASTORE SERVICE API.

METADATA

THE HIVE METASTORE SERVICE STORES THE METADATA FOR HIVE TABLES AND PARTITIONS

INFORMATION STORED IN THE METASTORE

- DATABASE IDS
- TABLE IDS
- INDEX IDS
- INDEX CREATION TIME
- TABLE CREATION TIME

LET'S SAY WE HAVE A TABLE WITH STUDENT DATA

THIS DATA SITS IN HDFS

HADOOP DISTRIBUTED FILE SYSTEM

HIVE TABLES
ARE STORED AS DIRECTORIES UNDER
HIVE'S WAREHOUSE DIRECTORY

HIVE'S WAREHOUSE DIRECTORY

THIS IS A DIRECTORY IN HDFS WHERE ALL THE HIVE DATA IS STORED

HIVE'S WAREHOUSE DIRECTORY

`hive.metastore.warehouse.dir`

THIS IS ONE OF THE PROPERTIES THAT'S SET DURING THE INITIAL CONFIGURATION OF HIVE

HIVE'S WAREHOUSE DIRECTORY

`hive.metastore.warehouse.dir`

THIS PROPERTY SHOULD BE SET TO THE HDFS PATH YOU WANT TO USE FOR STORING HIVE DATA

HIVE'S WAREHOUSE DIRECTORY

`hive.metastore.warehouse.dir`

THE DEFAULT LOCATION IS

`/user/hive/warehouse`

HIVE'S WAREHOUSE DIRECTORY

`/user/hive/warehouse`

WHEN WE CREATE A DATABASE IN HIVE,
IT BECOMES A NEW SUBDIRECTORY
UNDER THIS LOCATION

HIVE'S WAREHOUSE DIRECTORY

/user/hive/warehouse

```
create database students;
```

/user/hive/warehouse/students

HIVE'S WAREHOUSE DIRECTORY

/user/hive/warehouse

TABLES ARE STORED AS DIRECTORIES OR FOLDERS

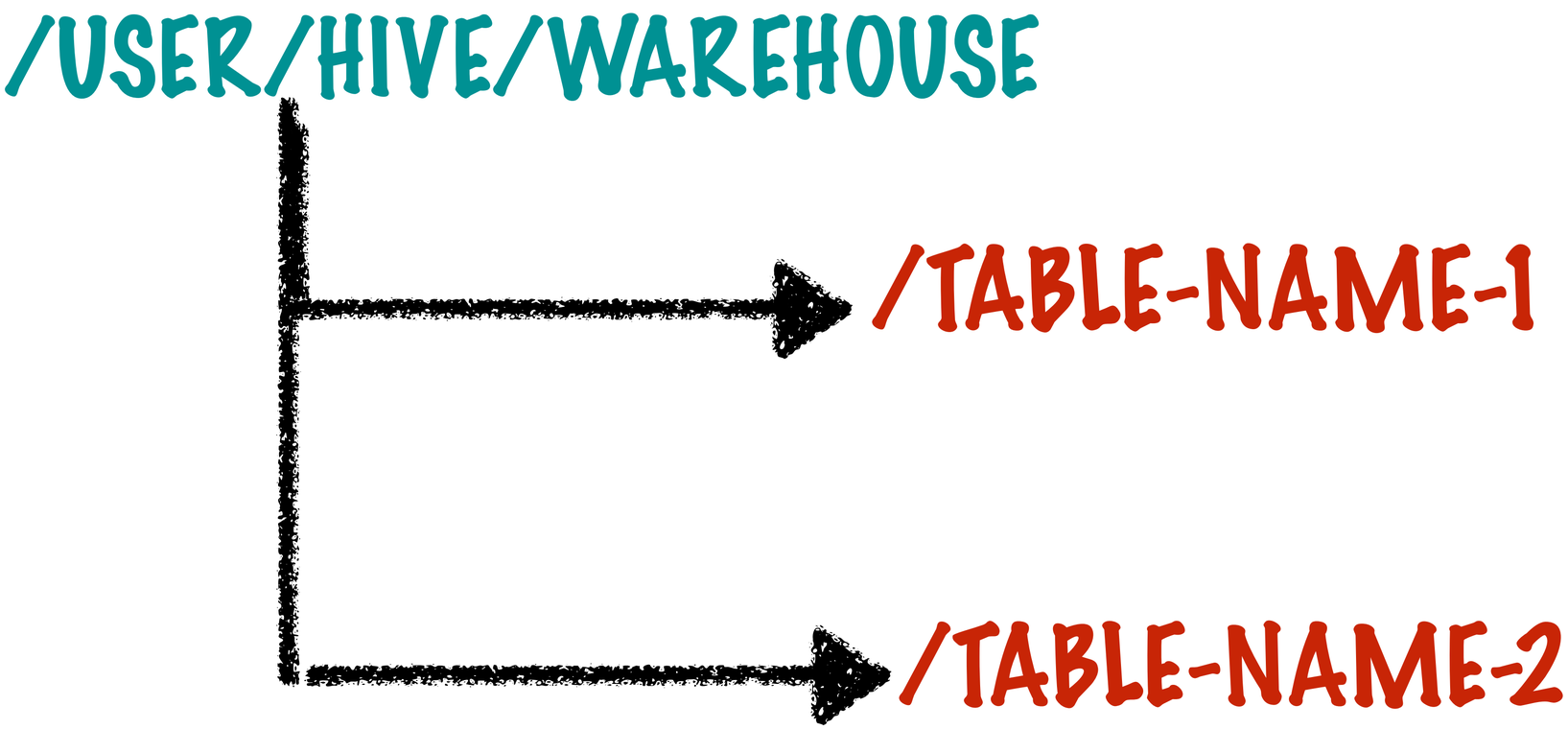
WHEN WE CREATE A TABLE IN HIVE, IT WILL BE A SUBDIRECTORY IN THE CORRESPONDING DATABASE DIRECTORY

HIVE'S WAREHOUSE DIRECTORY

`/user/hive/warehouse`

IF A DATABASE IS NOT SPECIFIED, THEN THE TABLE'S DIRECTORY IS CREATED DIRECTLY UNDER THE WAREHOUSE DIRECTORY

IF A DATABASE IS NOT SPECIFIED, THEN THE TABLE'S DIRECTORY IS CREATED DIRECTLY UNDER THE WAREHOUSE DIRECTORY



```
Navdeeps-MacBook-Pro:hadoop-2.7.2 navdeepsingh$  hadoop fs -ls  /user/hive/warehouse
16/05/19 21:37:59 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-ja
va classes where applicable
Found 16 items
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-17 17:40 /user/hive/warehouse/campus_housing
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-18 13:13 /user/hive/warehouse/movie_reviews_count
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-18 12:40 /user/hive/warehouse/movies
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-15 20:09 /user/hive/warehouse/pokes
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-18 13:58 /user/hive/warehouse/revenue_table
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-18 12:10 /user/hive/warehouse/reviews
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-18 13:13 /user/hive/warehouse/reviews_text
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-18 12:35 /user/hive/warehouse/reviews_w_partition
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-17 19:30 /user/hive/warehouse/sales_data_date_partition
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-18 15:45 /user/hive/warehouse/sales_data_date_product_partitio
n
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-18 14:24 /user/hive/warehouse/sales_data_without_partition
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-18 21:29 /user/hive/warehouse/stock_returns
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-19 05:06 /user/hive/warehouse/stock_returns_new
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-18 13:58 /user/hive/warehouse/stores_product_data
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-17 14:56 /user/hive/warehouse/students
drwxrwxr-x   - navdeepsingh supergroup          0 2016-05-19 04:29 /user/hive/warehouse/temporary
```

THE STUDENTS TABLE IS STORED AS A DIRECTORY

HIVE TABLES ARE STORED AS DIRECTORIES

DATA IS STORED IN FILES IN THESE DIRECTORIES

→ A TABLE CAN HAVE MULTIPLE FILES

```
Navdeeps-MacBook-Pro:hadoop-2.7.2 navdeepsingh$  hadoop fs -ls  /user/hive/warehouse/reviews
16/05/19 22:00:19 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-ja
va classes where applicable
Found 4 items
-rwxrwxr-x   1 navdeepsingh supergroup         92 2016-05-18 12:10 /user/hive/warehouse/reviews/000000_0
-rwxrwxr-x   1 navdeepsingh supergroup         69 2016-05-18 12:10 /user/hive/warehouse/reviews/000001_0
-rwxrwxr-x   1 navdeepsingh supergroup         88 2016-05-18 12:10 /user/hive/warehouse/reviews/000002_0
-rwxrwxr-x   1 navdeepsingh supergroup         87 2016-05-18 12:10 /user/hive/warehouse/reviews/000003_0
```

DATA IS STORED IN THESE FILES

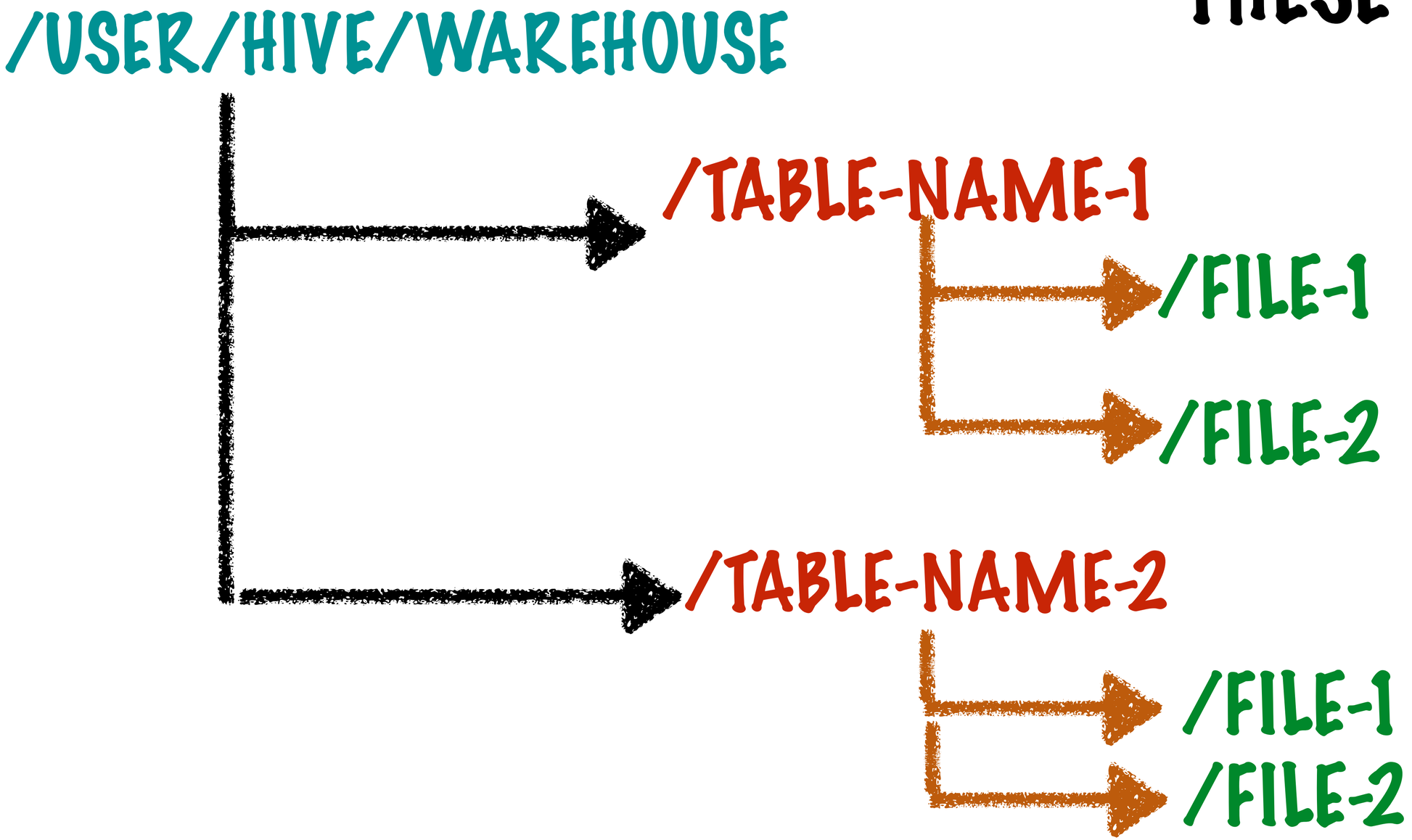


TABLE-1	COL 1	COL 2	COL 2
FILE-1 DATA	DATA	DATA	DATA
	DATA	DATA	DATA
FILE-2 DATA	DATA	DATA	DATA
	DATA	DATA	DATA

TABLE-2	COL 1	COL 2	COL 2
FILE-1 DATA	DATA	DATA	DATA
	DATA	DATA	DATA
FILE-2 DATA	DATA	DATA	DATA
	DATA	DATA	DATA

TABLES

THERE ARE TWO TYPES OF TABLES

MANAGED TABLES

EXTERNAL TABLES

TABLES

MANAGED TABLES

HIVE DATA IS MANAGED BY HIVE AND IS STORED IN ITS WAREHOUSE DIRECTORY

EXTERNAL TABLES

HIVE DATA EXISTS OUTSIDE THE WAREHOUSE DIRECTORY

BOTH TYPES OF TABLES KEEP METADATA IN THE METASTORE

The `'external'` keyword is used to create these external tables

TABLES

EXTERNAL TABLES

EXTERNAL TABLES ARE USED WHEN YOU WANT THE UNDERLYING DATA TO BE SHARED BETWEEN MULTIPLE PROGRAMS

EX: THE SAME FILES CAN BE USED BY HIVE, HBASE, PIG ETC

CREATING TABLES

CREATING TABLES

WE HAVE SEEN A TABLE BEFORE

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh	M	navdeep@loonycorn.com
4	Vitthal	Srinivasan	M	vitthal@loonycorn.com

THIS IS A TABLE NAMED 'STUDENTS'

HOW DO WE CREATE TABLES?

HOW DO WE CREATE TABLES?

THAT GETS US TO THE HIVE CREATE TABLE STATEMENT FOR A TABLE..

StudentID	FirstName	LastName	Gender	Email

HOW DO WE CREATE TABLES?

THAT GETS US TO THE HIVE CREATE TABLE STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email

```
CREATE TABLE Students
(
StudentID INT,
FirstName VARCHAR(30),
LastName VARCHAR(30),
Gender CHAR(1),
Email VARCHAR(30)
);
```

HOW DO WE CREATE TABLES?

THAT GETS US TO THE SQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email

```
CREATE TABLE Students
 (
StudentID INT,
FirstName VARCHAR(30),
LastName VARCHAR(30),
Gender CHAR(1),
Email VARCHAR(30)
);
```

START WITH THE NAME OF THE TABLE

HOW DO WE CREATE TABLES?

THAT GETS US TO THE SQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email

```
CREATE TABLE Students
(
    StudentID INT,
    FirstName VARCHAR(30),
    LastName VARCHAR(30),
    Gender CHAR(1),
    Email VARCHAR(30)
);
```

THEN SPECIFY EACH COLUMN NAME, ONE AT A TIME

HOW DO WE CREATE TABLES?

THAT GETS US TO THE SQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email

```
CREATE TABLE Students
(
StudentID  INT,
FirstName  VARCHAR(30),
LastName   VARCHAR(30),
Gender     CHAR(1),
Email      VARCHAR(30)
);
```

COLUMNS OF TABLES HAVE DATA TYPES

HOW DO WE CREATE TABLES?

THAT GETS US TO THE SQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email

```
CREATE TABLE Students
(
StudentID  INT,
FirstName  VARCHAR(30),
LastName   VARCHAR(30),
Gender     CHAR(1),
Email      VARCHAR(30)
);
```

SOME COLUMNS CONTAIN STRINGS

OTHERS CONTAIN NUMBERS

SOME COLUMNS CONTAIN STRINGS

OTHERS CONTAIN NUMBERS

COLUMNS OF TABLES HAVE DATA TYPES

THESE DATA TYPES ARE SPECIFIED WHEN THE TABLE IS CREATED

THESE DATA TYPES GOVERN HOW A COLUMN IS TREATED IN QUERIES

COLUMNS OF TABLES HAVE DATA TYPES

CHAR

VARCHAR

DECIMAL

DATETIME

INT

DATE

HOW DO WE CREATE TABLES?

THAT GETS US TO THE SQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email

```
CREATE TABLE Students
(
StudentID  INT,
FirstName  VARCHAR(30),
LastName   VARCHAR(30),
Gender     CHAR(1),
Email      VARCHAR(30)
);
```

COLUMNS OF TABLES HAVE DATA TYPES

IF WE WERE TO CREATE THE SAME TABLE IN RDBMS, WE WOULD HAVE LOTS OF OTHER FEATURES AT OUR DISPOSAL

```
CREATE TABLE Students
(
StudentID INT,
FirstName
VARCHAR(30),
LastName VARCHAR(30),
Gender CHAR(1),
Email VARCHAR(30)
);
```

```
CREATE TABLE Students
(
StudentID INT NOT NULL
AUTO_INCREMENT,
FirstName VARCHAR(30) NOT NULL,
LastName VARCHAR(30) NOT NULL,
Gender CHAR(1),
Email VARCHAR(30) NOT NULL,
PRIMARY KEY (StudentID)
)
```

IF WE WERE TO CREATE THE SAME TABLE IN RDBMS, WE WOULD HAVE LOTS OF OTHER FEATURES AT OUR DISPOSAL

HIVE DOESN'T SUPPORT CONSTRAINTS

```
CREATE TABLE Students
(
StudentID INT NOT NULL
AUTO_INCREMENT,
FirstName VARCHAR(30) NOT NULL,
LastName VARCHAR(30) NOT NULL,
Gender CHAR(1),
Email VARCHAR(30) NOT NULL,
PRIMARY KEY (StudentID)
)
```

EXERCISE: TABLE CREATION ONCE AGAIN

LET'S SAY WE HAVE A TABLE WITH SALES DATA

COLUMNS ARE NAMED 'STORELOCATION', 'PRODUCT', 'DATE','REVENUE'

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33
Bellandur	Nutella	January 18,2016	7,455.67
Bellandur	Peanut Butter	January 18,2016	5,316.89
Bellandur	Milk	January 18,2016	2,433.76
Koramangala	Bananas	January 18,2016	9,456.01
Koramangala	Nutella	January 18,2016	3,644.33
Koramangala	Peanut Butter	January 18,2016	8,988.64
Koramangala	Milk	January 18,2016	1,621.58

THIS IS A TABLE NAMED 'SALES_DATA'

THIS IS A TABLE NAMED 'SALES_DATA'

WHAT WOULD THE HIVESQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS LOOK LIKE?..

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33

THIS IS A TABLE NAMED 'SALES_DATA'

WHAT WOULD THE SQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS LOOK LIKE?..

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33

```
CREATE TABLE Sales_Data
(
StoreLocation VARCHAR(30),
Product VARCHAR(30),
OrderDate DATE,
Revenue DECIMAL(10,2)
);
```

THIS IS A TABLE NAMED 'SALES_DATA'

WHAT WOULD THE SQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS LOOK LIKE?..

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33

```
CREATE TABLE Sales_Data
(
StoreLocation VARCHAR(30),
Product VARCHAR(30),
OrderDate DATE,
Revenue DECIMAL(10,2)
);
```

THIS BIT IS PRETTY STANDARD..

THIS IS A TABLE NAMED 'SALES_DATA'

WHAT WOULD THE SQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS LOOK LIKE?..

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33

```
CREATE TABLE Sales_Data
(
StoreLocation VARCHAR(30),
Product VARCHAR(30),
OrderDate DATE,
Revenue DECIMAL(10,2)
);
```

BUT THIS IS INTERESTING!

THIS IS A TABLE NAMED 'SALES_DATA'

WHAT WOULD THE SQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS LOOK LIKE?..

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33

```
CREATE TABLE Sales_Data
(
StoreLocation VARCHAR(30),
Product VARCHAR(30),
OrderDate DATE,
Revenue DECIMAL(10,2)
);
```

DEC(10,2)

DEC(10,2)

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	**8,236.33**

THIS MEANS THIS COLUMN CAN HOLD NUMBERS WITH UPTO 10 DIGITS TOTAL WITH 2 DIGITS AFTER THE DECIMAL POINT

DEC(10,2)

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	**8,236.33**

THIS MEANS THIS COLUMN CAN HOLD NUMBERS WITH UPTO 10 DIGITS TOTAL WITH 2 DIGITS AFTER THE DECIMAL POINT

10 DIGITS! THAT'S A LOT OF BANANAS.

DEC(10,2)

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	**8,236.33**

THIS MEANS THIS COLUMN CAN HOLD NUMBERS WITH UPTO 10 DIGITS TOTAL WITH 2 DIGITS AFTER THE DECIMAL POINT

10 DIGITS! THAT'S A LOT OF BANANAS.

THAT'S PERFECT TO HOLD $ AND CENTS

THIS IS A TABLE NAMED 'SALES_DATA'

WHAT WOULD THE SQL CREATE TABLE STATEMENT FOR A TABLE LIKE THIS LOOK LIKE?..

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33

OUR FIRST DATE COLUMN!

```
CREATE TABLE Sales_Data
(
StoreLocation VARCHAR(30),
Product VARCHAR(30),
OrderDate DATE,
Revenue DECIMAL(10,2)
);
```

OUR FIRST DATE COLUMN!

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33

YOU CAN INSERT A DATE USING STRINGS (ENCLOSED IN SINGLE QUOTES) LIKE

`'January-18-2016'` `'Jan-18-2016'`

`'18-1-2016'` `'1-18-2016'`

YOU CAN CREATE A TABLE WITH THE SAME SCHEMA AS ANOTHER TABLE

```
CREATE TABLE
Sales_Data_Duplicate
like Sales_Data;
```

THIS WILL CREATE A TABLE WITH THE EXACT SAME SCHEMA AS SALES_DATA

```
CREATE TABLE
Sales_Data_Duplicate
like Sales_Data;
```

WHEN YOU TRY TO CREATE A TABLE, HIVE WILL THROW AN ERROR IF THE TABLE ALREADY EXISTS

```
CREATE TABLE if not exists
Sales_Data_Duplicate
like Sales_Data;
```

WITH THE ADDITION OF THIS CLAUSE, THE ERROR CAN BE AVOIDED

```
CREATE TABLE if not exists
Sales_Data_Duplicate
like Sales_Data;
```

THE TABLE IS CREATED ONLY IF ANOTHER TABLE WITH THE SAME NAME DOES NOT EXIST

```
CREATE TABLE if not exists
Sales_Data_Duplicate
like Sales_Data;
```

IF ANOTHER TABLE WITH THE SAME NAME EXISTS, THEN NOTHING IS DONE

```
CREATE TABLE if not exists
Sales_Data_Duplicate
like Sales_Data;
```

TABLES

THERE ARE TWO TYPES OF TABLES

MANAGED TABLES

BY DEFAULT WE CREATE MANAGED TABLES

EXTERNAL TABLES

LET'S SAY WE HAVE ANOTHER TABLE WITH ADDRESS DATA

COLUMNS ARE NAMED 'STUDENTID', 'DORMITORYNAME', 'APTNUMBER'

StudentID	DormitoryName	AptNumber
1	Gandhi House	110
2	Akbar Hall	231
3	Gandhi House	345
4	NULL	NULL

THIS IS A TABLE NAMED 'CAMPUS_HOUSING'

EXTERNAL TABLES TABLES

StudentID	DormitoryName	AptNumber
1	Gandhi House	110
2	Akbar Hall	231
3	Gandhi House	345
4	NULL	NULL

THIS IS A TABLE NAMED 'CAMPUS_HOUSING'

EXTERNAL TABLES

QUERY TO CREATE THIS TABLE WILL BE

```
CREATE External TABLE Campus_Housing_external
(
StudentID INT,
DormitoryName VARCHAR(50),
AptNumber INT
)
location‘/Users/navdeepsingh/Desktop/campus_housing';
```

StudentID	DormitoryName	AptNumber

EXTERNAL TABLES

```
CREATE External TABLE Campus_Housing_external
(
StudentID INT,
DormitoryName VARCHAR(50),
AptNumber INT
)
location'/Users/navdeepsingh/Desktop/campus_housing';
```

THE 'EXTERNAL' KEYWORD IS USED TO SIGNAL AN EXTERNAL TABLE

StudentID	DormitoryName	AptNumber

EXTERNAL TABLES

```
CREATE External TABLE Campus_Housing_external
(
StudentID INT,
DormitoryName VARCHAR(50),
AptNumber INT
)
location'/Users/navdeepsingh/Desktop/campus_housing'
```

THIS PART IS SIMILAR TO HOW WE CREATE MANAGED TABLES

StudentID	DormitoryName	AptNumber

EXTERNAL TABLES

```
CREATE External TABLE Campus_Housing_external
(
StudentID INT,
DormitoryName VARCHAR(50),
AptNumber INT
)
location'/users/navdeepsingh/campus_housing';
```

SPECIFY THE LOCATION WHERE YOU WANT THE HIVE DATA TO LIVE

StudentID	DormitoryName	AptNumber

EXTERNAL TABLES

```
CREATE External TABLE Campus_Housing_external
(
StudentID INT,
DormitoryName VARCHAR(50),
AptNumber INT
)
location'/users/navdeepsingh/campus_housing';
```

THIS LOCATION IS A DIRECTORY IN HDFS

```
Navdeeps-MacBook-Pro:hadoop-2.7.2 navdeepsingh$ hadoop fs -ls /users/navdeepsingh/
16/05/30 17:14:01 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-
va classes where applicable
Found 1 items
drwxr-xr-x   - navdeepsingh supergroup          0 2016-05-30 17:12 /users/navdeepsingh/campus_housing
```

TEMPORARY TABLES

TEMPORARY TABLES

TEMPORARY TABLES ARE THOSE TABLES WHICH GET DELETED AT THE END OF THE HIVE SESSION

TEMPORARY TABLES ARE USED TO STORE TEMPORARY DATA

TEMPORARY TABLES

TEMPORARY TABLES ARE THOSE TABLES WHICH GET DELETED AT THE END OF THE HIVE SESSION

MULTIPLE HIVE USERS CAN CREATE MULTIPLE HIVE TEMPORARY TABLES WITH THE SAME NAME BECAUSE EACH TABLE RESIDES IN A SEPARATE SESSION.

TEMPORARY TABLES

TEMPORARY TABLES ARE THOSE TABLES WHICH GET DELETED AT THE END OF THE HIVE SESSION

TEMPORARY TABLES DON'T SUPPORT PARTITIONED COLUMNS AND INDEXES

TEMPORARY TABLES

TEMPORARY TABLES ARE THOSE TABLES WHICH GET DELETED AT THE END OF THE HIVE SESSION

IF THERE ARE TWO TABLES WITH THE SAME NAME - ONE PERMANENT, ONE TEMPORARY

TEMPORARY TABLES

TEMPORARY TABLES ARE THOSE TABLES WHICH GET DELETED AT THE END OF THE HIVE SESSION

THE USER CAN'T ACCESS THE PERMANENT TABLE WITHOUT DROPPING THE TEMPORARY TABLE

TEMPORARY TABLES

THE USER CAN'T ACCESS THE
PERMANENT TABLE WITHOUT
DROPPING THE TEMPORARY TABLE

ALL THE REFERENCES MADE TO
THAT TABLE NAME WILL RESOLVE
TO THE TEMPORARY TABLE

HOW DO WE CREATE TEMPORARY TABLES?

BY USING THE 'TEMPORARY' KEYWORD

```
CREATE TEMPORARY TABLE Students
(
StudentID INT,
FirstName VARCHAR(30),
LastName VARCHAR(30),
Gender CHAR(1),
Email VARCHAR(30)
);
```

StudentID	FirstName	LastName	Gender	Email

HOW DO WE CREATE TEMPORARY TABLES?

BY USING 'TEMPORARY' KEYWORD

```
CREATE TEMPORARY TABLE Students
(
StudentID INT,
FirstName VARCHAR(30),
LastName VARCHAR(30),
Gender CHAR(1),
Email VARCHAR(30)
);
```

StudentID	FirstName	LastName	Gender	Email

THIS MAKES THE TABLE A TEMPORARY TABLE

HOW DO WE PUT STUFF INTO TABLES?

PERMANENT AND TEMPORARY

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastNam	Gender	Email
1	Vitthal	Srinivasan	M	vitthal@loonycorn.com

```
INSERT INTO TABLE Students
 (StudentID,Gender,Email,FirstName,LastName)
VALUES
(2,'F','swetha@loonycorn.com','Swetha','Kolalapudi')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com

```
INSERT INTO TABLE Students
 (StudentID,Gender,Email,FirstName,LastName)
VALUES
(2,'F','swetha@loonycorn.com','Swetha','Kolalapudi')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2				

```
INSERT INTO TABLE Students
 (StudentID,Gender,Email,FirstName,LastName)
VALUES
(2,'F','swetha@loonycorn.com','Swetha','Kolalapudi')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2			F	

```
INSERT INTO TABLE Students
 (StudentID,Gender,Email,FirstName,LastName)
VALUES
(2,'F','swetha@loonycorn.com','Swetha','Kolalapudi')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2			F	swetha@loonycorn.com

```
INSERT INTO TABLE Students
 (StudentID,Gender,Email,FirstName,LastName)
VALUES
(2,'F','swetha@loonycorn.com','Swetha','Kolalapudi')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha		F	swetha@loonycorn.com

```
INSERT INTO TABLE Students
 (StudentID,Gender,Email,FirstName,LastName)
VALUES
(2,'F','swetha@loonycorn.com','Swetha','Kolalapudi')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com

```
INSERT INTO TABLE Students
 (StudentID,Gender,Email,FirstName,LastName)
VALUES
(2,'F','swetha@loonycorn.com','Swetha','Kolalapudi')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com

```
INSERT INTO TABLE Students
 (StudentID,Gender,Email,FirstName,LastName)
VALUES
(2,'F','swetha@loonycorn.com','Swetha','Kolalapudi')
```

HOW DO WE PUT STUFF INTO TABLES?

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com

HOW DO WE PUT STUFF INTO TABLES?

COLUMN NAMES CAN EVEN BE SKIPPED ENTIRELY (BUT THEN YOU MUST INSERT IN ORDER!)

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

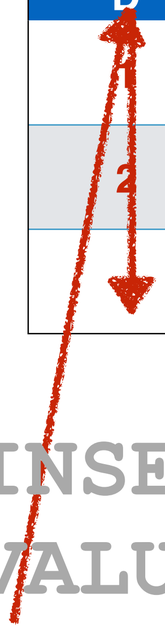
Stude	FirstNa	LastName	Gend	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com

```
INSERT INTO TABLE Students
VALUES
(3,'Navdeep','Singh','M','navdeep@loonycorn.com')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com



```
INSERT INTO TABLE Students
VALUES
(3,'Navdeep','Singh','M','navdeep@loonycorn.com')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3				

```
INSERT INTO TABLE Students
VALUES
(3,'Navdeep','Singh','M','navdeep@loonycorn.com')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

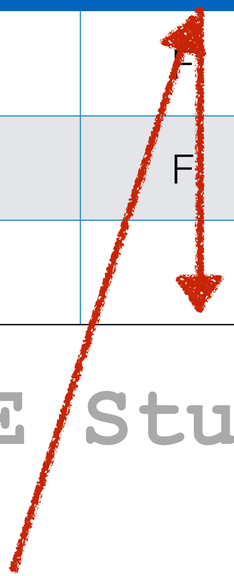
StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep			

```
INSERT INTO TABLE Students
VALUES
(3,'Navdeep','Singh','M','navdeep@loonycorn.com')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE SQL INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh		

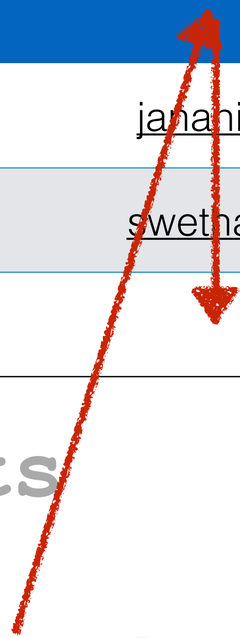


```
INSERT INTO TABLE Students
VALUES
(3,'Navdeep','Singh','M','navdeep@loonycorn.com')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh	M	



```
INSERT INTO TABLE Students
VALUES
(3,'Navdeep','Singh','M','navdeep@loonycorn.com')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh	M	navdeep@loonycorn.com

```
INSERT INTO TABLE Students
VALUES
(3,'Navdeep','Singh','M','navdeep@loonycorn.com')
```

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh	M	navdeep@loonycorn.com

```
INSERT INTO TABLE Students
VALUES
(3,'Navdeep','Singh','M','navdeep@loonycorn.com')
```

HOW DO WE PUT STUFF INTO TABLES?

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh	M	navdeep@loonycorn.com

HOW DO WE PUT STUFF INTO TABLES?

WE CAN PUT MULTIPLE RECORDS IN A TABLE AT ONE GO

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE INSERT STATEMENT FOR A TABLE LIKE THIS..

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh	M	navdeep@loonycorn.com

```
INSERT INTO TABLE Students
VALUES
(4,'Anu','Radha','F','anuradha@loonycorn.com'),
(5,'Vitthal','Srinivasan','M','vitthal@loonycorn.com'),
```

HOW DO WE PUT STUFF INTO TABLES?

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh	M	navdeep@loonycorn.com
4	Anu	Radha	F	anuradha@gmail.com
5	Vitthal	Srinivasan	M	vitthal@loonycorn.com

```
INSERT INTO TABLE Students
VALUES
(4,'Anu','Radha','F','anuradha@loonycorn.com'),
(5,'Vitthal','Srinivasan','M','vitthal@loonycorn.com');
```

HOW DO WE PUT STUFF INTO TABLES?

StudentID	FirstName	LastName	Gender	Email
1	Janani	Ravi	F	janani@loonycorn.com
2	Swetha	Kolalapudi	F	swetha@loonycorn.com
3	Navdeep	Singh	M	navdeep@loonycorn.com
4	Anu	Radha	F	anuradha@gmail.com
5	Vitthal	Srinivasan	M	vitthal@loonycorn.com

HOW DO WE PUT STUFF INTO TABLES?

WE CAN IMPORT DATA INTO THE TABLE FROM OTHER FILES

HOW DO WE PUT STUFF INTO TABLES?

WE CAN IMPORT DATA INTO THE TABLE FROM OTHER FILES

StudentID	DormitoryName	AptNumber

THIS IS A TABLE NAMED 'CAMPUS_HOUSING'

HOW DO WE PUT STUFF INTO TABLES?

WE CAN IMPORT DATA INTO THE TABLE FROM OTHER FILES

StudentID	DormitoryName	AptNumber

THIS IS A TABLE NAMED 'CAMPUS_HOUSING'

```
load data local inpath'/Users/navdeepsingh/
Desktop/campus_housing.txt'
overwrite into table Campus_Housing;
```

THAT GETS US TO THE OVERWRITE
STATEMENT FOR A SITUATION LIKE THIS..

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE OVERWRITE STATEMENT FOR A SITUATION LIKE THIS..

StudentID	DormitoryName	AptNumber

THIS PART INDICATES WHERE THE ORIGINAL DATA COMES FROM

```
load data local inpath '/Users/navdeepsingh/
Desktop/campus_housing.txt'
overwrite into table Campus_Housing;
```

THE DATA IS LOADED FROM A PARTICULAR FILE ON THE LOCAL MACHINE

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE OVERWRITE STATEMENT FOR A SITUATION LIKE THIS..

StudentID	DormitoryName	AptNumber

```
load data local inpath'/Users/navdeepsingh/
Desktop/campus_housing.txt'
overwrite into table Campus_Housing;
```

THIS IS THE SOURCE FILE PATH WHICH HOLDS THE DATA

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE OVERWRITE STATEMENT FOR A SITUATION LIKE THIS..

StudentID	DormitoryName	AptNumber

```
load data local inpath'/Users/navdeepsingh/
Desktop/campus_housing.txt'
overwrite into table Campus_Housing;
```

THE OVERWRITE KEYWORD MEANS THAT CONTENTS OF THE TARGET TABLE ARE COMPLETELY REPLACED BY CONTENTS OF THE FILE

HOW DO WE PUT STUFF INTO TABLES?

THAT GETS US TO THE OVERWRITE STATEMENT FOR A SITUATION LIKE THIS..

StudentID	DormitoryName	AptNumber

```
load data local inpath'/Users/navdeepsingh/
Desktop/campus_housing.txt'
overwrite into table Campus_Housing;
```

THE TABLE IN WHICH DATA IS INSERTED

HOW DO WE PUT STUFF INTO TABLES?

StudentID	DormitoryName	AptNumber

```
load data local inpath'/Users/navdeepsingh/Desktop/campus_housing.txt'
overwrite into table Campus_Housing;
```

WHEN WE LOAD DATA FROM A FILE, HIVE SIMPLY COPIES THAT FILE TO THE WAREHOUSE DIRECTORY

HOW DO WE PUT STUFF INTO TABLES?

```
load data local inpath'/Users/navdeepsingh/Desktop/campus_housing.txt'
overwrite into table Campus_Housing;
```

WHEN WE LOAD DATA FROM A FILE, HIVE SIMPLY COPIES THAT FILE TO THE WAREHOUSE DIRECTORY

```
Navdeeps-MacBook-Pro:hadoop-2.7.2 navdeepsingh$ hadoop fs -ls  /user/hive/warehouse/campus_housing
16/05/17 17:28:31 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rwxrwxr-x   1 navdeepsingh supergroup         72 2016-05-17 15:50 /user/hive/warehouse/campus_housing/campus_housing.txt
```

THE **CAMPUS_HOUSING.TXT** FILE IN THE HDFS WAREHOUSE DIRECTORY

HOW DO WE PUT STUFF INTO TABLES?

StudentID	DormitoryName	AptNumber

```
load data local inpath'/Users/navdeepsingh/
Desktop/campus_housing_2.txt'
into table Campus_Housing;
```

THIS IS THE SAME QUERY WITHOUT THE OVERWRITE KEYWORD

THIS APPENDS DATA TO THE TABLE

HOW DO WE PUT STUFF INTO TABLES?

StudentID	DormitoryName	AptNumber

```
load data local inpath'/Users/navdeepsingh/
Desktop/campus_housing_2.txt'
into table Campus_Housing;
```

THIS APPENDS THE CONTENT IN `CAMPUS_HOUSING_2.TEXT`

TO THE EXISTING CAMPUS_HOUSING TABLE

HOW DO WE PUT STUFF INTO TABLES?

```
load data local inpath'/Users/navdeepsingh/
Desktop/campus_housing_2.txt'
into table Campus_Housing;
```

THIS QUERY WITHOUT 'OVERWRITE' WILL ADD FILE

CAMPUS_HOUSING_2.TEXT

TO THE DIRECTORY OF THE TABLE CAMPUS_HOUSING IN HIVE'S WAREHOUSE DIRECTORY

```
16/05/17 17:41:19 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rwxrwxr-x   1 navdeepsingh supergroup         72 2016-05-17 15:50 /user/hive/warehouse/campus_housing/campus_housing.txt
-rwxrwxr-x   1 navdeepsingh supergroup        114 2016-05-17 17:40 /user/hive/warehouse/campus_housing/campus_housing_2.txt
```

HOW DO WE PUT STUFF INTO TABLES?

ALL OF THIS MEANS THAT

YOU CAN IMPORT DATA FROM FILES JUST BY COPYING THEM TO THE RIGHT LOCATION IN HDFS

HOW DO WE PUT STUFF INTO TABLES?

WHEN YOU READ DATA FROM THE TABLE

HDFS WILL READ THE FILES IN THE TABLE'S DIRECTORY

HOW DO WE PUT STUFF INTO TABLES?

AN IMPORTANT DETAIL:

HIVE EXPECTS THE FILES IN THE TABLE'S DIRECTORY TO BE IN A SPECIFIC FORMAT

HOW DO WE PUT STUFF INTO TABLES?

AN IMPORTANT DETAIL:

IF THE FORMAT DOES NOT MATCH, HIVE WILL NOT BE ABLE TO READ THE DATA FROM THE FILE

HOW DO WE PUT STUFF INTO TABLES?

AN IMPORTANT DETAIL:

THIS FORMAT CAN BE CHANGED TO MATCH YOUR FILE USING THE ALTER STATEMENT

EXAMPLE:

```
alter table Campus_Housing
set SERDEPROPERTIES ('field.delim'=',');
```

HOW DO WE PUT STUFF INTO TABLES?

EXAMPLE:

```
alter table Campus_Housing
set SERDEPROPERTIES ('field.delim'=',');
```

THIS WILL CHANGE THE PROPERTIES OF CAMPUS_HOUSING

HOW DO WE PUT STUFF INTO TABLES?

EXAMPLE:

```
alter table Campus_Housing
set SERDEPROPERTIES ('field.delim'=',');
```

SERDEPROPERTIES IS A DICT THAT TELLS HIVE HOW TO READ THE FILES IN THE TABLE DIRECTORY

HOW DO WE PUT STUFF INTO TABLES?

EXAMPLE:

```
alter table Campus_Housing
set SERDEPROPERTIES ('field.delim'=',');
```

THIS TELLS HIVE THAT THE FILES FOR THIS TABLE ARE COMMA SEPARATED

HOW DO WE PUT STUFF INTO TABLES?

WE CAN IMPORT DATA INTO A TABLE FROM OTHER TABLES

LET'S SAY WE HAVE A TABLE WITH SALES DATA

COLUMNS ARE NAMED 'STORELOCATION', 'PRODUCT', 'DATE','REVENUE'

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33
Bellandur	Nutella	January 18,2016	7,455.67
Bellandur	Peanut Butter	January 18,2016	5,316.89
Bellandur	Milk	January 18,2016	2,433.76
Koramangala	Bananas	January 18,2016	9,456.01
Koramangala	Nutella	January 18,2016	3,644.33
Koramangala	Peanut Butter	January 18,2016	8,988.64
Koramangala	Milk	January 18,2016	1,621.58

THIS IS A TABLE NAMED 'SALES_DATA'

HOW DO WE PUT STUFF INTO TABLES?

WE CAN IMPORT DATA INTO A TABLE FROM OTHER TABLES

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33
Bellandur	Nutella	January 18,2016	7,455.67
Bellandur	Peanut Butter	January 18,2016	5,316.89
Bellandur	Milk	January 18,2016	2,433.76
Koramangala	Bananas	January 18,2016	9,456.01
Koramangala	Nutella	January 18,2016	3,644.33
Koramangala	Peanut Butter	January 18,2016	8,988.64
Koramangala	Milk	January 18,2016	1,621.58

THIS TABLE'S DATA CAN BE USED TO CREATE NEW TABLES

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33
Bellandur	Nutella	January 18,2016	7,455.67
Bellandur	Peanut Butter	January 18,2016	5,316.89
Bellandur	Milk	January 18,2016	2,433.76
Koramangala	Bananas	January 18,2016	9,456.01
Koramangala	Nutella	January 18,2016	3,644.33
Koramangala	Peanut Butter	January 18,2016	8,988.64
Koramangala	Milk	January 18,2016	1,621.58

WE WANT TO CREATE TABLE WHICH STORES PRODUCTS ALONG WITH THE STORES WHERE THEY ARE AVAILABLE

StoreLocation	Product
Bellandur	Bananas
Bellandur	Nutella
Bellandur	Peanut Butter
Bellandur	Milk
Koramangala	Bananas
Koramangala	Nutella
Koramangala	Peanut Butter
Koramangala	Milk

WE WANT TO CREATE TABLE WHICH STORES PRODUCTS ALONG WITH THE STORES WHERE THEY ARE AVAILABLE

CREATE A TABLE FIRST

```
CREATE TABLE stores_product_data
(
StoreLocation VARCHAR(30),
Product VARCHAR(30)
);
```

StoreLocation	Product
Bellandur	Bananas
Bellandur	Nutella
Bellandur	Peanut Butter
Bellandur	Milk
Koramangala	Bananas
Koramangala	Nutella
Koramangala	Peanut Butter
Koramangala	Milk

StoreLocation	Product	Date	Revenue

INSERT INTO THAT TABLE

StoreLocation	Product
Bellandur	Bananas
Bellandur	Nutella
Bellandur	Peanut Butter
Bellandur	Milk
Koramangala	Bananas
Koramangala	Nutella
Koramangala	Peanut Butter
Koramangala	Milk

```
insert overwrite table stores_product_data
select StoreLocation,Product
from Sales_Data;
```

StoreLocation	Product	Date	Revenue
Bellandur	Nutella	January 18,2016	7,455.67

INSERT INTO THAT TABLE

StoreLocation	Product
Bellandur	Bananas
Bellandur	Nutella
Bellandur	Peanut Butter
Bellandur	Milk
Koramangala	Bananas
Koramangala	Nutella
Koramangala	Peanut Butter
Koramangala	Milk

```
insert overwrite table stores_product_data
select StoreLocation,Product
from Sales_Data;
```

THE OVERWRITE KEYWORD CAUSES THE CONTENTS OF THE TARGET TABLE TO BE COMPLETELY REPLACED BY THE RESULT OF THE SELECT STATEMENT

INSERT INTO THAT TABLE

THIS IS THE SOURCE OF THE DATA

StoreLocation	Product
Bellandur	Bananas
Bellandur	Nutella
Bellandur	Peanut Butter
Bellandur	Milk
Koramangala	Bananas
Koramangala	Nutella
Koramangala	Peanut Butter
Koramangala	Milk

```
insert overwrite table stores_product_data
select StoreLocation,Product
from Sales_Data;
```

THE SELECT STATEMENT IS USED TO FETCH DATA FROM TWO COLUMNS OF THE SALES DATA TABLE

HOW DO WE PUT STUFF INTO TABLES?

SO FAR THIS WAS:

SOURCE TABLE: ONE

DESTINATION TABLE: ONE

HOW DO WE PUT STUFF INTO TABLES?

BUT WE CAN HAVE

SOURCE TABLE: ONE

DESTINATION TABLE: TWO OR MORE

(MULTITABLE INSERT)

HOW DO WE PUT STUFF INTO TABLES?

WE CAN INSERT DATA INTO TWO DIFFERENT TABLES FROM ONE SOURCE TABLE IN ONE COMMAND

(MULTITABLE INSERT)

LET'S SAY WE HAVE A TABLE WITH SALES DATA

StoreLocation	Product	Date	Revenue
Bellandur	Bananas	January 18,2016	8,236.33
Bellandur	Nutella	January 18,2016	7,455.67
Bellandur	Peanut Butter	January 18,2016	5,316.89
Bellandur	Milk	January 18,2016	2,433.76
Koramangala	Bananas	January 18,2016	9,456.01
Koramangala	Nutella	January 18,2016	3,644.33
Koramangala	Peanut Butter	January 18,2016	8,988.64
Koramangala	Milk	January 18,2016	1,621.58

THIS IS A TABLE NAMED 'SALES_DATA'

THIS WILL BE OUR SOURCE TABLE

WE WILL INSERT DATA FROM THIS TABLE TO STORES_PRODUCT_DATA

CREATE A TABLE FIRST

```
CREATE TABLE stores_product_data
(
StoreLocation VARCHAR(30),
Product VARCHAR(30)
);
```

StoreLocation	Product
Bellandur	Bananas
Bellandur	Nutella
Bellandur	Peanut Butter
Bellandur	Milk
Koramangala	Bananas
Koramangala	Nutella
Koramangala	Peanut Butter
Koramangala	Milk

StoreLocation	Product	Date	Revenue

CREATE ANOTHER TABLE TO HOLD PRODUCT DATA

CREATE A TABLE FIRST

```
CREATE TABLE product_data
(
Product VARCHAR(30)
);
```

Product
Bananas
Nutella
Peanut Butter
Milk

StoreLocation	Product	Date	Revenue

INSERT INTO BOTH TABLES

StoreLocation	Product
Bellandur	Bananas
Bellandur	Nutella
Bellandur	Peanut Butter
Bellandur	Milk
Koramangala	Bananas
Koramangala	Nutella
Koramangala	Peanut Butter
Koramangala	Milk

Product
Bananas
Nutella
Peanut Butter
Milk

```
from Sales_Data
insert overwrite table stores_product_data
select StoreLocation,Product
insert overwrite table product_data
select distinct Product;
```

StoreLocation	Product	Date	Revenue
Bellandur	Nutella	January 18,2016	7,455.67

INSERT INTO BOTH TABLES

```
from Sales_Data
insert overwrite table stores_product_data
select StoreLocation,Product
insert overwrite table product_data
select distinct Product;
```

THERE ARE TWO INSERT COMMANDS
ONE FOR EACH TABLE

StoreLocation	Product	Date	Revenue

INSERT INTO BOTH TABLES

```
from Sales_Data
insert overwrite table stores_product_data
select StoreLocation,Product
insert overwrite table product_data
select distinct Product;
```

THE SOURCE IS ONE SINGLE TABLE
FOR BOTH THE INSERT STATEMENTS

StoreLocation	Product	Date	Revenue

INSERT INTO BOTH TABLES

```
from Sales_Data
insert overwrite table stores_product_data
select StoreLocation,Product
insert overwrite table product_data
select distinct Product;
```

INSERT STATEMENT TO PUT DATA INTO TABLE STORES_PRODUCT_DATA

StoreLocation	Product	Date	Revenue

INSERT INTO BOTH TABLES

```
from Sales_Data
insert overwrite table stores_product_data
select StoreLocation,Product
insert overwrite table product_data
select distinct Product;
```

THE OVERWRITE KEYWORD REPLACES ALL THE EXISTING DATA IN THAT TABLE

StoreLocation	Product	Date	Revenue

INSERT INTO BOTH TABLES

```
from Sales_Data
insert overwrite table stores_product_data
select StoreLocation,Product
insert overwrite table product_data
select distinct Product;
```

ALL STORE LOCATIONS AND PRODUCTS ARE SELECTED FROM THE SOURCE TABLE

StoreLocation	Product	Date	Revenue

INSERT INTO BOTH TABLES

```
from Sales_Data
insert overwrite table stores_product_data
select StoreLocation,Product
insert overwrite table product_data
select distinct Product;
```

THE SECOND INSERT STATEMENT TO PUT DATA INTO THE PRODUCT_DATA TABLE

StoreLocation	Product	Date	Revenue

INSERT INTO BOTH TABLES

```
from Sales_Data
insert overwrite table stores_product_data
select StoreLocation,Product
insert overwrite table product_data
select distinct Product;
```

THE OVERWRITE KEYWORD WILL REPLACE ALL DATA IN THE DESTINATION TABLE

StoreLocation	Product	Date	Revenue

INSERT INTO BOTH TABLES

```
from Sales_Data
insert overwrite table stores_product_data
select StoreLocation,Product
insert overwrite table product_data
select distinct Product;
```

THE DISTINCT KEYWORD CAN BE USED TO RETURN ONLY UNIQUE PRODUCT VALUES

StoreLocation	Product	Date	Revenue

THE DISTINCT KEYWORD CAN BE USED TO RETURN ONLY DISTINCT (DIFFERENT) VALUES

StoreLocation	Product	Date	Revenue
Bellandur	Nutella	January 18,2016	7,455.67
Bellandur	Peanut Butter	January 18,2016	5,316.89
Bellandur	Milk	January 18,2016	2,433.76
Koramangala	Bananas	January 18,2016	9,456.01
Koramangala	Nutella	January 18,2016	3,644.33
Koramangala	Peanut Butter	January 18,2016	8,988.64
Koramangala	Milk	January 18,2016	1,621.58

```
select distinct Product;
```

REMOVES DUPLICATES

Product
Bananas
Nutella
Peanut Butter
Milk

ALTERING TABLES

ALTERING TABLES

HIVE ALOWS US TO CHANGE THE STRUCTURE OF THE TABLE

CHANGE THE NUMBER OF COLUMNS, THE DATATYPES OF COLUMNS ETC.

ALTERING TABLES

HIVE ALOWS US TO CHANGE THE STRUCTURE OF THE TABLE

THIS IS DONE BY USING THE 'ALTER' COMMAND

ALTERING TABLES

THIS IS DONE BY USING THE 'ALTER' COMMAND

```
alter table old_table_name new_table_name;
```

TO RENAME TABLE

ALSO CHANGES TABLE'S DIRECTORY NAME IN HIVE'S WAREHOUSE DIRECTORY

```
alter table
revenue_per_day rename
to revenue_table;
```

ALTERING TABLES

THIS IS DONE BY USING THE 'ALTER' COMMAND

```
alter table table_name add columns(col_name data_type);
```

TO ADD COLUMN 'COL_NAME' OF 'DATA_TYPE'

```
alter table
revenue_table add
columns (new_date DATE);
```

ALTERING TABLES

THIS IS DONE BY USING THE 'ALTER' COMMAND

```
CREATE TABLE
revenue_per_day
(
OrderDate DATE,
Revenue DECIMAL(10,2),
new_date DATE
);
```

WE WANT TO DELETE THE NEW_DATE COLUMN

ALTERING TABLES

```
CREATE TABLE revenue_per_day
(
OrderDate DATE,
Revenue DECIMAL(10,2),
new_date DATE
);
```

```
alter table revenue_table
replace columns(OrderDate DATE, Revenue DECIMAL(10,2))
```

WE USE REPLACE COMMAND WITH ALTER TO DELETE A COLUMN

ALTERING TABLES

```
CREATE TABLE revenue_per_day
(
OrderDate DATE,
Revenue DECIMAL(10,2),
new_date DATE
);
```

```
alter table revenue_table
replace columns(OrderDate DATE, Revenue DECIMAL(10,2))
```

WE USE REPLACE COMMAND WITH ALTER TO DELETE A COLUMN

TO DELETE A COLUMN DON'T PUT IT IN THE REPLACE COLUMNS LIST

DELETING TABLES

DELETING TABLES

HIVE ALLOWS US TO DELETE TABLES

THIS IS DONE BY USING THE 'DROP' COMMAND

DELETING TABLES

THIS IS DONE BY USING THE 'DROP' COMMAND

```
DROP table table_name;
```

```
drop table revenue_per_day;
```

THIS DELETES THE TABLE REVENUE_PER_DAY

DELETING DATA INSIDE TABLES

DELETING DATA INSIDE TABLES

HIVE ALLOWS US TO DELETE THE TABLE'S DATA WITHOUT DELETING THE TABLE ITSELF

THIS IS DONE BY USING THE 'TRUNCATE' COMMAND

DELETING DATA INSIDE TABLES

HIVE ALOWS US TO DELETE THE TABLE'S DATA WITHOUT DELETING THE TABLE ITSELF

THIS IS DONE BY USING THE 'TRUNCATE' COMMAND

```
truncate table table_name;
```

```
truncate table revenues;
```

THIS COMMAND DELETES ALL THE DATA INSIDE THE REVENUES TABLE

DELETING AND UPDATING A SINGLE ROW IN A TABLE

DELETING AND UPDATING A SINGLE ROW IN A TABLE

HIVE DOES NOT PROVIDE A WAY TO PERFORM THESE 2 OPERATIONS!

DELETING AND UPDATING A SINGLE ROW IN A TABLE

HIVE DOES NOT PROVIDE A WAY TO PERFORM THESE 2 OPERATIONS!

HIVE PRETENDS THAT THE UNDERLYING DATA IS IN THE FORM OF A TABLE BUT IT IS A BATCH PROCESSING SYSTEM AT HEART

DELETING AND UPDATING A SINGLE ROW IN A TABLE

HIVE DOES NOT PROVIDE A WAY TO PERFORM THESE 2 OPERATIONS!

HIVE HAS MASSIVE OVERHEADS IN JOB SUBMISSION AND SCHEDULING

DELETING AND UPDATING A SINGLE ROW IN A TABLE

HIVE DOES NOT PROVIDE A WAY TO PERFORM THESE 2 OPERATIONS!

HIVE DOES NOT OFFER REAL-TIME QUERIES AND ROW LEVEL UPDATES